## Roadmap to Microsoft Excel Documentation

| If you are new to spreadsheets or to Microsoft Excel | If you are familiar with Lotus 1-2-3 | For all users | |
|---|---|---|---|
| ■ | ■ | ■ | For instructions on installing and starting Microsoft Excel, see ***Getting Started.*** |
| ■ | | | For a hands-on tutorial, see "Learning Microsoft Excel" in ***Getting Started.*** |
| ■ | | | For a set of online lessons about Microsoft Excel, see the **Microsoft Excel Tutorial.** In Microsoft Excel for Windows or OS/2, start the tutorial by choosing Tutorial from the Microsoft Excel Help menu; in Microsoft Excel for the Macintosh, start the tutorial by double-clicking the Microsoft Excel Tutorial icon in the Microsoft Excel folder. |
| | | ■ | For complete information on using Microsoft Excel, including procedures, tips, and troubleshooting, see the ***User's Guide.*** |
| | ■ | | To learn how Microsoft Excel differs from Lotus 1-2-3 and how to convert files, see Chapter 2, "Using Lotus 1-2-3 with Microsoft Excel" in the ***User's Guide.*** |
| | | ■ | For a description of the sample worksheet models and macros included on your Microsoft Excel disks, see Appendix B, "Microsoft Excel Library" in the ***User's Guide.*** |
| | | ■ | For complete descriptions of all worksheet and macro functions, see the ***Function Reference.*** |
| | ■ | ■ | For instant access to reference information on your computer screen, use online **Help.** In Microsoft Excel for Windows or OS/2, press F1 or choose Index from the Microsoft Excel Help menu; in Microsoft Excel for the Macintosh, choose Help from the Window menu. |
| ■ | | | For a guide to the Microsoft Excel screen, see the ***Quick Reference.*** |
| | | ■ | For a list of keyboard shortcuts and capsule descriptions of worksheet functions, see the ***Quick Reference.*** |
| | ■ | | For a conversion table of Lotus 1-2-3 commands, see the ***Quick Reference.*** |

# Microsoft® Excel

## Function Reference

*Spreadsheet with Business Graphics and Database*

*Version 3.0*
*Apple® Macintosh® Series, OS/2 Series, or Windows™ Series*

*Microsoft Corporation*

Information in this document is subject to change without notice and does not represent a commitment on the part of Microsoft Corporation. The software and/or databases described in this document are furnished under a license agreement or nondisclosure agreement. The software and/or databases may be used or copied only in accordance with the terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license or nondisclosure agreement. The purchaser may make one copy of the software for backup purposes. No part of this manual and/or databases may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or information storage and retrieval systems, for any purpose other than the purchaser's personal use, without the express written permission of Microsoft Corporation.

© 1990, 1991 Microsoft Corporation. All rights reserved. Printed in the United States of America.

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of Commercial Computer Software—Restricted Rights at 48 CFR 52.227-19, as applicable. Contractor/Manufacturer is Microsoft Corporation, One Microsoft Way, Redmond, Washington 98052-6399.

Microsoft and the Microsoft logo are registered trademarks of Microsoft Corporation. Autosum, *Making it all make sense*, Toolbar, and Windows are trademarks of Microsoft Corporation.

IBM is a registered trademark of International Business Machines Corporation.

HP and LaserJet are registered trademarks and NewWave is a trademark of Hewlett-Packard Company.

PostScript is a registered trademark of Adobe Systems, Incorporated.

Macintosh is a registered trademark and AppleEvents is a trademark of Apple Computer, Incorporated.

Microsoft documentation uses the term "OS/2" to refer to the OS/2 systems—Microsoft® Operating System/2 (MS® OS/2) and IBM® OS/2. Similarly, the term "DOS" refers to both the MS-DOS® and IBM Personal Computer DOS operating systems. The name of a specific operating system is used when it is necessary to note features that are unique to that system.

This manual was produced using Microsoft Word.

Document Number AB14242-0291
OEM W/P725-3Z

# Table of Contents

***Welcome. . .lx***

***About Functions. . .lx***

Two Types of Functions. . .ix
Worksheet Functions. . .ix
Macro Functions. . .ix
How Function Descriptions Are Organized. . .xi
Using Arguments. . .xii

***Functions Listed by Category or Action. . .xv***

***Differences Due to Operating Environment. . .xxvi***

***Function Reference. . .1***

A1.R1C1. . .1
ABS. . .1
ABSREF. . .1
ACOS. . .2
ACOSH. . .2
ACTIVATE. . .2
ACTIVATE.NEXT. . .3
ACTIVATE.PREV. . .3
ACTIVE.CELL. . .3
ADD.ARROW. . .4
ADD.BAR. . .4
ADD.COMMAND. . .5
ADD.MENU. . .6
ADD.OVERLAY. . .7
ADDRESS. . .7
ALERT. . .8
ALIGNMENT. . .9
AND. . .9
APP.ACTIVATE. . .10
APP.MAXIMIZE. . .11
APP.MINIMIZE. . .11
APP.MOVE. . .11
APP.RESTORE. . .12
APP.SIZE. . .12
APPLY.NAMES. . .13
APPLY.STYLE. . .14
AREAS. . .14
ARGUMENT. . .15
ARRANGE.ALL. . .16
ASIN. . .16
ASINH. . .17
ASSIGN.TO.OBJECT. . .17
ATAN. . .17
ATAN2. . .18
ATANH. . .18
ATTACH.TEXT. . .18
ATTRIBUTES. . .19
AVERAGE. . .20
AXES. . .20
BEEP. . .21
BORDER. . .21
BREAK. . .22
BRING.TO.FRONT. . .22
CALCULATE.DOCUMENT. . .23
CALCULATE.NOW. . .23
CALCULATION. . .23
CALL. . .24
CALLER. . .24
CANCEL.COPY. . .25
CANCEL.KEY. . .25
CELL. . .26
CELL.PROTECTION. . .28
CHANGE.LINK. . .28
CHAR. . .29
CHECK.COMMAND. . .29
CHOOSE. . .30

CLEAN. . .30
CLEAR. . .31
CLOSE. . .31
CLOSE.ALL. . .32
CODE. . .32
COLOR.PALETTE. . .32
COLUMN. . .33
COLUMN.WIDTH. . .33
COLUMNS. . .34
COMBINATION. . .35
CONSOLIDATE. . .35
COPY. . .36
COPY.CHART. . .36
COPY.PICTURE. . .37
COS. . .37
COSH. . .37
COUNT. . .38
COUNTA. . .38
CREATE.NAMES. . .39
CREATE.OBJECT. . .40
CREATE.PUBLISHER. . .41
CUSTOM.REPEAT. . .42
CUSTOM.UNDO. . .42
CUT. . .43
DATA.DELETE. . .43
DATA.FIND. . .43
DATA.FIND.NEXT. . .43
DATA.FIND.PREV. . .43
DATA.FORM. . .43
DATA.SERIES. . .44
DATE. . .44
DATEVALUE. . .45
DAVERAGE. . .46
DAY. . .46
DAYS360. . .46
DCOUNT. . .47
DCOUNTA. . .47
DDB. . .47
DEFINE.NAME. . .48
DEFINE.STYLE. . .49
DELETE.ARROW. . .51
DELETE.BAR. . .51
DELETE.COMMAND. . .51
DELETE.FORMAT. . .52
DELETE.MENU. . .52
DELETE.NAME. . .53
DELETE.OVERLAY. . .53
DELETE.STYLE. . .53
DEMOTE. . .54
DEREF. . .54
Dfunction. . .54
DGET. . .56
DIALOG.BOX. . .57
DIRECTORY. . .58
DISABLE.INPUT. . .58
DISPLAY. . .59
DMAX. . .60
DMIN. . .60
DOCUMENTS. . .60
DOLLAR. . .61
DPRODUCT. . . 61
DSTDEV. . .61
DSTDEVP. . .62
DSUM. . .62
DUPLICATE. . .62
DVAR. . .62
DVARP. . .62
ECHO. . .62
EDIT.COLOR. . .62
EDIT.DELETE. . .63
EDIT.REPEAT. . .64
EDIT.SERIES. . .64
EDITION.OPTIONS. . .65
ELSE. . .66
ELSE.IF. . .66
EMBED. . .67



ENABLE.COMMAND. . .67
END.IF. . .68
ERROR. . .68
EXACT. . .69
EXEC. . .70
EXECUTE. . .71
EXP. . .72
EXTRACT. . .72
FACT. . .73
FALSE. . .73
FCLOSE. . .73
FILE.CLOSE. . .74
FILE.DELETE. . .74
FILES. . .75
FILL.DOWN. . .76
FILL.LEFT. . .76
FILL.RIGHT. . .76
FILL.UP. . .76
FILL.WORKGROUP. . .76
FIND. . .76
FIXED. . .77
FONT. . .78
FOPEN. . .78
FOR. . .79
FOR.CELL. . .80
FORMAT.FONT. . .80
FORMAT.LEGEND. . .82
FORMAT.MAIN. . .82
FORMAT.MOVE. . .84
FORMAT.NUMBER. . .85
FORMAT.OVERLAY. . .85
FORMAT.SIZE. . .87
FORMAT.TEXT. . .88
FORMULA. . .89
FORMULA.ARRAY. . .90
FORMULA.CONVERT. . .91
FORMULA.FILL. . .91
FORMULA.FIND. . .92
FORMULA.FIND.NEXT. . .93
FORMULA.FIND.PREV. . .93
FORMULA.GOTO. . .93
FORMULA.REPLACE. . .94
FPOS. . .95
FREAD. . .96
FREADLN. . .97
FREEZE.PANES. . .97
FSIZE. . .98
FULL. . .98
FV. . .98
FWRITE. . .99
FWRITELN. . .99
GALLERY.3D.AREA. . .100
GALLERY.3D.COLUMN. . .100
GALLERY.3D.LINE. . .100
GALLERY.3D.PIE. . .100
GALLERY.AREA. . .100
GALLERY.BAR. . .101
GALLERY.COLUMN. . .101
GALLERY.LINE. . .101
GALLERY.PIE. . .102
GALLERY.SCATTER. . .102
GET.BAR. . .102
GET.CELL. . .103
GET.CHART.ITEM. . .105
GET.DEF. . .106
GET.DOCUMENT. . .107
GET.FORMULA . . .110
GET.LINK.INFO. . .110
GET.NAME. . .111
GET.NOTE. . .112
GET.OBJECT. . .112
GET.WINDOW. . .115
GET.WORKSPACE. . .116
GOAL.SEEK. . .118
GOTO. . .118
GRIDLINES. . .119

GROUP. . .119
GROWTH. . .119
HALT. . .120
HELP. . .121
HIDE. . .122
HIDE.OBJECT. . .122
HLINE. . .122
HLOOKUP. . .123
HOUR. . .123
HPAGE. . .124
HSCROLL. . .124
IF. . .125
INDEX. . .127
INDIRECT. . .129
INFO. . .129
INITIATE. . .130
INPUT. . .131
INSERT. . .133
INT. . .133
IPMT. . .133
IRR. . .134
IS Functions. . .135
JUSTIFY. . .136
LAST.ERROR. . .137
LEFT. . .137
LEGEND. . .137
LEN. . .137
LINEST. . .138
LINKS. . .142
LIST.NAMES. . .142
LN. . .143
LOG. . .143
LOG10. . .144
LOGEST. . .144
LOOKUP. . .145
LOWER. . .147
MAIN.CHART. . .147
MAIN.CHART.TYPE. . .147
MATCH. . .148
MAX. . .149
MDETERM. . .149
MEDIAN. . .150
MERGE.STYLES. . .150
MESSAGE. . .151
MID. . .151
MIN. . .152
MINUTE. . .152
MINVERSE. . .153
MIRR. . .154
MMULT. . .155
MOD. . .155
MONTH. . .156
MOVE. . .156
N. . .157
NA. . .157
NAMES. . .158
NEW. . .159
NEW.WINDOW. . .159
NEXT. . .160
NOT. . .160
NOTE. . .160
NOW. . .161
NPER. . .161
NPV. . .162
OBJECT.PROTECTION. . .163
OFFSET. . .163
ON Functions. . .164
ON.DATA. . .164
ON.KEY. . .165
ON.RECALC. . .167
ON.TIME. . .168
ON.WINDOW. . .168
OPEN. . .169
OPEN.LINKS. . .170
OPEN.MAIL. . .171
OR. . .172



OUTLINE. . .173
OVERLAY. . .173
OVERLAY.CHART.TYPE. . .173
PAGE.SETUP. . .173
PARSE. . .175
PASTE. . .176
PASTE.LINK. . .176
PASTE.PICTURE. . .177
PASTE.PICTURE.LINK. . .177
PASTE.SPECIAL. . .177
PATTERNS. . .179
PI. . .182
PLACEMENT. . .182
PMT. . .183
POKE. . .183
PPMT. . .184
PRECISION. . .185
PREFERRED. . .185
PRINT. . .186
PRINT.PREVIEW. . .186
PRINTER.SETUP. . .187
PRODUCT. . .187
PROMOTE. . .187
PROPER. . .188
PROTECT.DOCUMENT. . .188
PV. . .189
QUIT. . .191
RAND. . .191
RATE. . .192
REFTEXT. . .192
REGISTER. . .193
RELREF. . .194
REMOVE.PAGE.BREAK. . .195
RENAME.COMMAND. . .195
REPLACE. . .196
REPLACE.FONT. . .196
REPT. . .197
REQUEST. . .197
RESTART. . .198
RESULT. . .198
RETURN. . .200
RIGHT. . .200
ROUND. . .200
ROW. . .201
ROW.HEIGHT. . .201
ROWS. . .202
RUN. . .202
SAVE. . .203
SAVE.AS. . .204
SAVE.NEW.OBJECT. . .205
SAVE.WORKSPACE. . .205
SCALE. . .206
SEARCH. . .208
SECOND. . .209
SELECT. . .209
SELECT.CHART. . .212
SELECT.END. . .212
SELECT.LAST.CELL. . .213
SELECT.PLOT.AREA. . .213
SELECT.SPECIAL. . .213
SELECTION. . .214
SEND.KEYS. . .214
SEND.MAIL. . .215
SEND.TO.BACK. . .216
SERIES. . .216
SET.CRITERIA. . .217
SET.DATABASE. . .217
SET.EXTRACT. . .217
SET.NAME. . .217
SET.PAGE.BREAK. . .218
SET.PREFERRED. . .218
SET.PRINT.AREA. . .218
SET.PRINT.TITLES. . .219
SET.UPDATE.STATUS. . .219
SET.VALUE. . .220
SHARE. . .221

SHARE.NAME. . .221
SHORT.MENUS. . .222
SHOW.ACTIVE.CELL. . .222
SHOW.BAR. . .222
SHOW.CLIPBOARD. . .223
SHOW.DETAIL. . .223
SHOW.INFO. . .224
SHOW.LEVELS. . .224
SIGN. . .225
SIN. . .225
SINH. . .225
SIZE. . .225
SLN. . .226
SORT. . .226
SPLIT. . .227
SQRT. . .228
STANDARD.FONT. . .228
STDEV. . .228
STDEVP. . .229
STEP. . .229
STYLE. . .230
SUBSCRIBE.TO. . .230
SUBSTITUTE. . .231
SUM. . .231
SUMPRODUCT. . .232
SYD. . .232
T. . .233
TABLE. . .233
TAN. . .234
TANH. . .234
TERMINATE. . .234
TEXT. . .234
TEXT.BOX. . .235
TEXTREF. . .235
TIME. . .236
TIMEVALUE. . .237
TODAY. . .237
TRANSPOSE. . .237
TREND. . .238
TRIM. . .239
TRUE. . .239
TRUNC. . .239
TYPE. . .240
UNDO. . .241
UNGROUP. . .241
UNHIDE. . .241
UNLOCKED.NEXT. . .241
UNLOCKED.PREV. . .241
UNREGISTER. . .242
UPDATE.LINK. . .242
UPPER. . .242
VALUE. . .243
VAR. . .243
VARP. . .244
VDB. . .244
VIEW.3D. . .245
VLINE. . .246
VLOOKUP. . .246
VOLATILE. . .247
VPAGE. . .247
VSCROLL. . .247
WAIT. . .248
WEEKDAY. . .249
WHILE. . .249
WINDOWS. . .250
WORKGROUP. . .250
WORKSPACE. . .251
YEAR. . .251

***Appendix A Using the CALL and REGISTER Functions. . .253***

Data Types in Microsoft Excel for Windows or OS/2. . .253

Data Types in Microsoft Excel for the Macintosh. . .255

***Index. . .259***

# Welcome

Welcome to the *Microsoft Excel Function Reference*, a complete description of Microsoft Excel's built-in worksheet and macro functions. This book is for users of Microsoft® Excel for Windows™, Microsoft® Excel for OS/2, and Microsoft® Excel for the Macintosh.

The *Function Reference* is divided into three parts. The first part reviews the differences between worksheet and macro functions, explains the conventions used in this manual, and shows how the information for each function is organized.

The second part lists functions by category or by the action they perform and points out functions that are available only in specific operating environments.

The third part describes all of Microsoft Excel's functions in alphabetical order.

### If You Are New to Functions

If you are new to functions, first read "About Functions" in this section. To learn how to enter worksheet functions into formulas or how to create and use macros, see the *Microsoft Excel User's Guide*.

### If You Are Experienced with Functions

If you are experienced with functions, read "How Function Descriptions Are Organized" later in this section.

# About Functions

Functions are calculation tools that allow you to perform decision-making, action-taking, and value-returning operations automatically. Microsoft Excel offers a wide variety of functions that perform many different types of calculations.

## Two Types of Functions

Microsoft Excel has two types of functions: worksheet functions and macro functions. You can use worksheet functions on both worksheets and macro sheets. For example, the worksheet function COS returns the cosine of an angle; you usually use COS on a worksheet, but you can also use it on a macro sheet.

Macro functions are available only on macro sheets and are instructions you use in macros to perform actions. Many macro functions also accept and return values. Functions on a macro sheet are calculated only when the associated macro is run. Macro functions are identified in this book by the words "Macro Sheets Only" in the bar above the function name.

## Worksheet Functions

You use worksheet functions on worksheets and macro sheets to calculate and return values based on information you provide. Worksheet functions take a value or values, perform some operation on them, and return a value or values. Functions on a worksheet are calculated as soon as you enter them. The 11 categories of worksheet functions are listed later in this section under "Worksheet Functions by Category."

### Online Help for Worksheet Functions

Microsoft Excel includes a directory of worksheet functions as part of the online Help system. To use the online directory in Microsoft Excel for Windows or OS/2, choose the Help Index command or press F1. To use the online directory in Microsoft Excel for the Macintosh, choose the Window Help command. After you choose the command, select Functions from the list of Help topics.

## Macro Functions

Microsoft Excel's macro language is made up of more than 300 macro functions. You can use macros to automate anything you can do manually with Microsoft Excel, such as choosing commands, selecting options in dialog boxes, entering data on worksheets, and changing the current selection. You can also use the macro language to create custom functions that perform specialized calculations.

## Macro Functions That Perform Actions

There are four kinds of macro functions that perform actions:

| Category | Action |
|---|---|
| Command-equivalent functions | Choose commands |
| Dialog-box functions | Display dialog boxes |
| Other action-equivalent functions | Perform keyboard or mouse actions (other than choosing commands) |
| Customizing functions | Customize Microsoft Excel |

### Command-equivalent functions

Carrying out a command-equivalent function is the same as choosing a particular command from one of the Microsoft Excel menus. For example, carrying out the FILE.DELETE function is like choosing the File Delete command. The arguments to a command-equivalent function correspond to options, check boxes, text boxes, and list boxes associated with that command.

***Option buttons*** are represented by number values. For example, in the Format Alignment dialog box, five option buttons represent the five alignment options. In the ALIGNMENT macro function, you select one of the options by specifying a number from 1 to 5 corresponding to those buttons.

***Check boxes*** are represented by logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box. If an argument is omitted, the current state of the check box is not changed.

***Text boxes*** are represented by text values. Text arguments should be enclosed in quotation marks

***List boxes*** are represented by number or text values, and allow you to make a choice from a list of items in a list box. You either specify a number (for example, to select a color in the Options Display dialog box) or text (for example, to select a format in the Format Number dialog box).

### Dialog-box functions

For every command that displays a dialog box, there is a dialog-box function. The dialog-box form has the same name as the command-equivalent function, but its name is followed by a question mark. This form of a function displays a dialog box while the macro is running. The dialog box remains on the screen until you select the options you want. After you close the dialog box by choosing OK or Cancel, the macro continues to run. In dialog-box forms of functions, all arguments are optional.

### Other action-equivalent functions

Other action-equivalent functions correspond to actions you take without choosing a command, such as selecting a cell (the SELECT function), scrolling through a window (the HSCROLL and VSCROLL functions), or using a tool or button on the tool bar (for example, the PROMOTE and DEMOTE functions).

### Customizing functions

Customizing functions let you perform actions that can be performed only by macros, such as creating custom menus, commands, and dialog boxes.

## Other Macro Functions

There are two kinds of macro functions that do not perform actions: control functions and value-returning functions.

### Control functions

Control functions allow you to direct how your command macros run. You can use control functions to create loops, redirect macro execution to other cells, and control how Microsoft Excel receives and returns information from functions. For example, you can use the FOR and NEXT functions to create a loop to repeat an operation a specified number of times.

### Value-returning functions

Value-returning functions return or manipulate information about almost every detail of your Microsoft Excel working environment. You can use them to get information about cells, objects, windows, documents, and many other Microsoft Excel items. You can design macros that decide what actions to perform based on the values returned by these functions.



## How Function Descriptions Are Organized

To help you find information, function descriptions follow a template consisting of the following parts:

- **Name** The name of the function as you type it into Microsoft Excel, including all required and optional arguments in the correct order (that is, the syntax).
- **Return value** What the function returns, or what command or action the function performs, and suggestions for using the function.
- **Arguments** A description of each argument, in the order of the function's syntax.
- **General information** General information about the function.
- **Notes** Notes and tips to make using the function easier.
- **Examples** Examples showing how to use the function.
- **Related functions** A list of related functions with short descriptions.

## Conventions in This Manual

- In headings, required arguments are ***italic and bold*** and optional arguments are *italic* but not bold.
- In text, all arguments are *italic* whether they are required or optional. See "Syntax of Functions" later in this section for more information on required and optional arguments.
- Argument names use underline characters to represent spaces between words; for example, *num_chars* is an argument name.
- Macro and cell names use no punctuation between words; for example, SalesTotals is a cell name.
- In procedures, anything you type is shown in **bold.**
- Functions are shown without the equal sign. Remember to type an equal sign at the beginning of every formula, but not before functions in nested formulas.

***Note***

The illustrations in this book may look slightly different from your screen, depending on the type of monitor and computer you are using.

## Syntax of Functions

Every function description begins with a main heading that shows the function's syntax. For example, the heading for the LEFT function looks like this:

***LEFT(text,num_chars)***

In the heading, the function's name and required arguments are bold; optional arguments are not bold. If you do not provide required arguments to a function, you may not be able to enter the function in a cell. In the example above, the argument *text* is bold, and therefore it is required. *Num_chars* is not bold and therefore is optional, so either of the following formulas is allowed:

```
LEFT("Hiram",2)
LEFT("Conley")
```

The formula LEFT( ) is not allowed, because *text* is a required argument.

When an argument is followed by an ellipsis (...), you can have more than one argument of that data type. You can have up to 14 arguments for some functions, as long as the total number of characters in the formula does not exceed 1024. Also, no individual string can be larger than 255 characters. For example, here is the syntax for the MAX function:

***MAX(number1,number2,...)***

Any of the following formulas would be allowed:

```
MAX(26)
MAX(26,31)
MAX(26,31,29)
```

Functions with an empty set of parentheses after the name take no arguments, although you must include the parentheses in order for Microsoft Excel to recognize the function.

In descriptive text, all argument names appear in plain italics (not bold) so that it's easier to see them. For example, the CELL function has one required

argument and one optional argument. In the heading, its syntax is shown as:

***CELL(info_type,reference)***

When we talk about the *info_type* and *reference* arguments in text, however, both arguments are italic and not bold. Use the format in the heading to determine whether an argument is required or optional.

### Argument Names

Many argument names tell you what kind of information to use as the argument. For example, in the function LEFT(*text,num_chars*), the first argument must be text and the second argument must be a number.

If the abbreviation *num*, *ref*, or *log* appears in the name of the argument, that argument must be a number, a reference, or a logical value, respectively. Similarly, the words *number*, *reference*, *logical*, *text*, and *array* in the name of an argument specify that the argument must be of that data type. *Value* means that the argument can be anything that results in a single value. That value can be a number, text, logical, or error value.

#### Note

Not every argument name specifies the argument's data type. For functions with many arguments, adding the data-type abbreviations makes the syntax listing too long to read easily. For these functions, the data type for each argument is specified in the argument description.

## Using Arguments

Arguments are the information that a function uses to produce a new value or perform an action. Arguments are always located to the right of the function's name and are enclosed in parentheses. Most arguments are expected to be of a certain data type. The argument you give should be either a value of the appropriate type or a value of another type that Microsoft Excel can convert to the appropriate type. For more information on how Microsoft Excel converts argument types, see "Converting Data Types" later in this section.

An argument can be anything that produces a desired data type. For example, the SUM function, which adds its arguments, can take from 1 to 14 arguments. You can give the SUM function any of the following four kinds of arguments, which produce a number or numbers:

- A value that is a number, such as:
  SUM(1,10,100)
- A formula that results in a number, such as:
  SUM(0.5+0.5,AVERAGE(5,5),10^2)

  Using a function as an argument to a function, as in the preceding example, is called nesting functions. In that example, the AVERAGE function is an argument to the SUM function. You can nest up to eight levels of functions in a formula.
- A reference to a cell or range of cells that contains a number or a formula that results in a number, such as:
  SUM(A1,A2)
  SUM(A1:A5)

  The second example above is equivalent to the formula SUM(A1,A2,A3,A4,A5). An advantage of using the range form is that the argument A1:A5 counts as only one argument, while the other form counts as five arguments. If you wanted to add more than 14 numbers, you would have to use the range form, because you can only use 14 arguments with any function.
- A name that refers to a value, a formula, or a cell or range containing a value or formula, such as:
  SUM(CurrentRate,Inflation)

### Types of Arguments

A function can have from 0 to 14 arguments. These arguments can be:

- Numbers
- Text
- Logical values
- Error values

***Numbers*** For example, 5.003, 0, 150.286, or –30. Numbers without decimals are called integers; 5, 0, 150, and –30 are integers, for example. Numbers are accurate to 15 digits.

***Text*** For example, "a", "Word", "w/punc.", or "". Text values used in formulas must be enclosed in double quotation marks. If the text itself contains quotation marks, use two double quotation marks for each double quotation mark in the text. For example, to find the length in characters for this text— in the "good" old days—enter the formula:

```
LEN("in the ""good"" old days")
```

Text values can be a maximum of 255 characters long including quotation marks. A text constant that contains no characters is written as "" and is sometimes called "empty text."

#### Note

If text used as an argument is not enclosed in quotation marks, Microsoft Excel assumes it is a name and tries to translate the value that the name refers to. If the unquoted text is not a name, and therefore has no associated value, Microsoft Excel returns the #NAME? error value.

***Logical values*** The two logical values are TRUE and FALSE. Logical arguments can also be statements, such as B10>20, that evaluate to TRUE or FALSE.

***Error values*** The error values are #DIV/0!, #N/A, #NAME?, #NULL!, #NUM!, #REF!, and #VALUE!.

You can also use the following types of arguments to provide one of the four basic argument types (numbers, text, logical values, and error values) to a function.

***References*** For example, $A$10, A10, $A10, A$10, R1C1, or R[10]C[–10]. References can refer to single cells, ranges, or multiple selections, and can be relative, absolute, or mixed. When you use a reference as an argument that is supposed to be a number, text, error value, or logical value, the contents of the cells specified by the reference are used as the argument. For information on references in macro functions, see Chapter 19, "Designing and Writing a Command Macro," in the *Microsoft Excel User's Guide*.

#### Notes

- Functions that return references display the value contained in the reference instead of the reference. To use the reference itself in a macro formula, use the REFTEXT function to convert the reference to a text string.
- To supply a multiple selection as a single reference argument, enclose the reference in a second set of parentheses, for example:
  SELECT(("R1C1:R2C2,R3C3:R4C4"))

***Names*** For example, the name Sales could be defined to refer to a cell that contains a value, to a formula that returns a value, or to a constant number. These types of values would be displayed in the Refers To box in the Formula Define Name dialog box as follows:

| Value | Sample display |
|---|---|
| Cell reference | =$A$3 |
| Formula | =SUM($A$1:$A$3) |
| Number | =100000 |
| Text | ="Sales" |
| Error value | =#VALUE! |
| Logical value | =TRUE |

Each time you use a name as an argument to a function, Microsoft Excel evaluates the value the name refers to and uses that value to calculate the function.

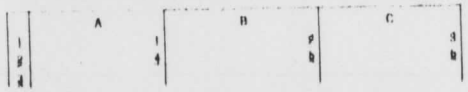
If a name refers to a reference, and the function is not expecting a reference, Microsoft Excel evaluates the reference and returns its contents as the argument.

***Formulas*** You can use formulas, which can return numbers, text, errors, logical values, or arrays, as arguments to functions. When a formula is used as an argument, the formula is evaluated and its result is used by the function.

***Arrays*** Microsoft Excel converts arrays in three ways:

- Some worksheet functions (AND, AVERAGE, COUNT, COUNTA, MAX, MIN, OR, STDEV, STDEVP, SUM, VAR, and VARP) can take from 1 to 14 arguments, all of the same data type. If an array is used as an argument to one of those functions, each element of the array produces a single result. For example,

SUM({1,2,3;4,5,6}) has one element and produces one result, 21. This operates just as if you had entered the formula SUM(A1:C2) on the following worksheet:




- If the formula is an array formula, the function acts on each element of the array and produces an array result. Each element of the array is evaluated as if it were a separate and unique value used as the argument. For example, the ABS function takes a single number as an argument and returns the absolute value of the number. You can use an array as the argument to ABS: the array formula {=ABS({1,–2,3;–4,5,–6})} equals {1,2,3;4,5,6}.
- If the formula is not an array formula and an array argument is not expected, the function takes the first element of the array and ignores the rest. For example, if the following formula is not entered as an array formula, then: ABS({1,–2,3;–4,5,–6}) equals 1.

If the formula requires an array as the argument and you use a single value, Microsoft Excel creates an array from the value. For example, the following formula compares a single value, the text SALES.XLS, with a three-item array returned by the DOCUMENTS function:

```
OR("SALES.XLS"=DOCUMENTS())
```

To evaluate the formula, Microsoft Excel converts the single value SALES.XLS into an array of the same size as the one returned by the DOCUMENTS function, as shown below:

```
OR({"SALES.XLS","SALES.XLS","SALES.XLS"}={"Chart1",
"Macro1","SALES.XLS"})
```

The two arrays shown in the function are in effect evaluated as follows:

| | |
|---|---|
| SALES.XLS = Chart1 | returns FALSE |
| SALES.XLS = Macro1 | returns FALSE |
| SALES.XLS = SALES.XLS | returns TRUE |

Microsoft Excel evaluates each element in the first array against the corresponding element in the second array. Since one of the formulas in the example evaluates to TRUE, and since the OR function returns TRUE if any of its arguments returns TRUE, the function returns TRUE.

For more information about arrays, see Chapter 7, "Analyzing and Calculating a Worksheet," in the *Microsoft Excel User's Guide*.

### Tips

If you are entering a nested formula, using the Formula Paste Function command to enter the functions can help you get the parentheses in the right place. It's best to avoid nesting formulas so deeply that they become difficult to read and edit, however.

If you work with a lot of complicated formulas, you might want to learn about custom functions. For information on custom functions, see Chapter 17, "Creating and Using Custom Worksheet Functions," in the *Microsoft Excel User's Guide*.

## Converting Data Types

If you use an argument that does not produce the correct data type for a function, Microsoft Excel tries to convert the argument to the required data type. For example, the LEN function takes one text argument and returns the number of characters in that argument; LEN("abc") is equal to 3. If you enter the formula LEN(1202) in a cell, Microsoft Excel internally converts the number 1202 to the text value "1202". The formula LEN(1202) is therefore equal to 4.

Numbers, text, and logical values are converted as follows:

| If an argument should be | Result |
|---|---|
| A number, but is given as text | Microsoft Excel tries to convert the text to a number. If the text is in any standard number, date, time, or currency format, it is interpreted as if the quotation marks weren't there, just as in the VALUE function. |
| A number, but is given as a logical value | TRUE is converted to 1, and FALSE is converted to 0. |
| Text, but is given as a number | The number is treated as text (as if it were enclosed in quotation marks). |
| Text, but is given as a logical value | TRUE is converted to "TRUE", and FALSE is converted to "FALSE". |
| Logical, but is given as a number | Any number other than 0 is TRUE, and 0 is FALSE. |
| Logical, but is given as text | Only the text "TRUE" and "FALSE" are converted. Any other text is ignored. |



### Using Commas in a List of Arguments

You must separate individual arguments with commas, but be careful not to type extra commas. If you use a comma to hold a place for an argument, but you don't enter the argument, Microsoft Excel substitutes the default value for that argument, unless the argument is required. For example, for a worksheet function that takes three arguments, if you enter (,*arg2*,*arg3*) as the arguments, Microsoft Excel substitutes an appropriate value for *arg1*. If you enter (*arg1*,,), it substitutes appropriate values for *arg2* and *arg3*.

In worksheet functions, particularly in functions that count the number of arguments before calculating, extra commas will affect the count of arguments and therefore the way the function is calculated. For example, AVERAGE(1,2,3,4,5) returns 3, but AVERAGE(,,1,2,3,4,5) returns 2.14. In macro functions, extra commas act as placeholders for arguments and generally have no effect.

For most arguments, the value substituted for an omitted argument is 0, FALSE, or "" (empty text), depending on what the data type of the argument should be. For omitted *reference* arguments, the default value is usually the active cell or selection.

For example, in the DISPLAY function (equivalent to the Options Display command), if you omit the first three arguments and specify TRUE for the fourth, the state of the first three check boxes in the Options Display dialog box remains unchanged. If an omitted argument is assumed to be some other value, the description of the argument will tell you so.

### Note

If you are using a reference as an argument and that reference uses a comma as a union operator, enclose the reference in parentheses. The AREAS function, for example, takes one argument: a reference. If you try to enter the formula AREAS(A1,C1), Microsoft Excel interprets A1 and C1 as two separate arguments and displays the "Error in formula" message. The correct form would be AREAS((A1,C1)). For information on reference operators, see Chapter 3, "Creating and Using a Worksheet," in the *Microsoft Excel User's Guide*.

# Functions Listed by Category or Action

## Worksheet Functions by Category

This section lists worksheet functions by the following categories:

- Database
- Date and time
- Financial
- Information
- Logical
- Lookup
- Mathematical
- Matrix
- Statistical
- Text
- Trigonometric

### Database

DAVERAGE(*database,field,criteria*)
Returns the average of selected database entries

DCOUNT(*database,field,criteria*)
Counts the cells containing numbers from a specified database and criteria

DCOUNTA(*database,field,criteria*)
Counts nonblank cells from a specified database and criteria

DGET(*database,field,criteria*)
Extracts from a database a single record that matches the specified criteria

DMAX(*database,field,criteria*)
Returns the maximum value from selected database entries

DMIN(*database,field,criteria*)
Returns the minimum value from selected database entries

DPRODUCT(*database,field,criteria*)
Multiplies the values in a particular field of records that match the criteria in a database

DSTDEV(*database,field,criteria*)
Estimates the standard deviation based on a sample of selected database entries

DSTDEVP(*database,field,criteria*)
Calculates the standard deviation based on the entire population of selected database entries

DSUM(*database,field,criteria*)
Adds numbers in a database

DVAR(*database,field,criteria*)
Estimates variance based on a sample from selected database entries

DVARP(*database,field,criteria*)
Calculates variance based on the entire population of selected database entries

### Date and time

DATE(*year,month,day*)
Returns the serial number of a specified date

DATEVALUE(*date_text*)
Converts a date in the form of text to a serial number

DAY(*serial_number*)
Converts a serial number to a day

DAYS360(*start_date,end_date*)
Calculates the number of days between two dates based on a 360-day year

HOUR(*serial_number*)
Converts a serial number to an hour

MINUTE(*serial_number*)
Converts a serial number to a minute

MONTH(*serial_number*)
Converts a serial number to a month

NOW()
Returns the serial number of the current date and time

SECOND(*serial_number*)
Converts a serial number to a second

TIME(*hour,minute,second*)
Returns the serial number of a particular time

TIMEVALUE(*time_text*)
Converts a time in the form of text to a serial number

TODAY()
Returns the serial number of today's date

WEEKDAY(*serial_number*)
Converts a serial number to a day of the week

YEAR(*serial_number*)
Converts a serial number to a year

### Financial

DDB(*cost,salvage,life,period,factor*)
Returns the depreciation of an asset for a specified period using the double-declining balance method

FV(*rate,nper,pmt,pv,type*)
Returns the future value of an investment

IPMT(*rate,per,nper,pv,fv,type*)
Returns the interest payment for an investment for a given period

IRR(*values,guess*)
Returns the internal rate of return for an investment, without financing costs or reinvestment gains

MIRR(*values,finance_rate,reinvest_rate*)
Returns the internal rate of return where positive and negative cash flows are financed at different rates



NPER(*rate,pmt,pv,fv,type*)
Returns the number of payments for an investment

NPV(*rate,value1,value2,...*)
Returns the net present value of an investment based on cash flows that do not have to be constant

PMT(*rate,nper,pv,fv,type*)
Returns the periodic total payment for an investment

PPMT(*rate,per,nper,pv,fv,type*)
Returns the payment on the principal for an investment for a given period

PV(*rate,nper,pmt,fv,type*)
Returns the present value of an investment

RATE(*nper,pmt,pv,fv,type,guess*)
Returns the interest rate per period of an investment

SLN(*cost,salvage,life*)
Returns the straight-line depreciation of an asset for one period

SYD(*cost,salvage,life,per*)
Returns the sum-of-years' digits depreciation of an asset for a specified period

VDB(*cost,salvage,life,start_period,end_period,factor,no_switch*)
Returns the depreciation of an asset for a specified or partial period using a declining balance method

### Information

ADDRESS(*row_num,column_num,abs_num,a1,sheet_text*)
Returns a reference as text to a single cell in a worksheet

AREAS(*reference*)
Returns the number of areas in a reference

CELL(*info_type,reference*)
Returns information about the formatting, location, or contents of a cell

COLUMN(*reference*)
Returns the column number of a reference

COLUMNS(*array*)
Returns the number of columns in a reference

INDIRECT(*ref_text,a1*)
Returns a reference indicated by a text value

INFO(*type_num*)
Returns information about the current operating environment

ISBLANK(*value*)
Returns TRUE if the value is blank

ISERR(*value*)
Returns TRUE if the value is any error value except #N/A

ISERROR(*value*)
Returns TRUE if the value is any error value

ISLOGICAL(*value*)
Returns TRUE if the value is a logical value

ISNA(*value*)
Returns TRUE if the value is the error value #N/A

ISNONTEXT(*value*)
Returns TRUE if the value is not text

ISNUMBER(*value*)
Returns TRUE if the value is a number

ISREF(*value*)
Returns TRUE if the value is a reference

ISTEXT(*value*)
Returns TRUE if the value is text

N(*value*)
Returns the value converted to a number

NA()
Returns the error value #N/A

OFFSET(*reference,rows,cols,height,width*)
Returns a reference offset from a given reference

ROW(*reference*)
Returns the row number of a reference

ROWS(*array*)
Returns the number of rows in a reference

T(*value*)
Converts its arguments to text

TYPE(*value*)
Returns a number indicating the data type of a value

### Logical

AND(*logical1,logical2,...*)
Returns TRUE if all its arguments are TRUE

FALSE()
Returns the logical value FALSE

IF(*logical_test,value_if_true,value_if_false*)
Specifies a logical test to perform

NOT(*logical*)
Reverses the logic of its argument

OR(*logical1,logical2,...*)
Returns TRUE if any argument is TRUE

TRUE()
Returns the logical value TRUE

### Lookup

CHOOSE(*index_num,value1,value2,...*)
Chooses a value from a list of values

HLOOKUP(*lookup_value,table_array, row_index_num*)
Looks in the first row of an array and moves down the column to return the value of a cell

INDEX(*reference,row_num,column_num, area_num*)
INDEX(*array,row_num,column_num*)
Uses an index to choose a value from a reference or array

LOOKUP(*lookup_value,lookup_vector, result_vector*)
LOOKUP(*lookup_value,array*)
Looks up values in a reference or array

MATCH(*lookup_value,lookup_array,match_type*)
Looks up values in a reference or array

VLOOKUP(*lookup_value,table_array, col_index_num*)
Looks in the first column of an array and moves across the row to return the value of a cell

### Mathematical

ABS(*number*)
Returns the absolute value of a number

EXP(*number*)
Returns e raised to a given number

FACT(*number*)
Returns the factorial of a number

INT(*number*)
Rounds a number down to the nearest integer

LN(*number*)
Returns the natural logarithm of a number

LOG(*number,base*)
Returns the logarithm of a number to a specified base

LOG10(*number*)
Returns the base-10 logarithm of a number

MOD(*number,divisor*)
Gives the remainder from division

PI()
Returns the value π

PRODUCT(*number1,number2,...*)
Multiplies its arguments

RAND()
Returns a random number between 0 and 1

ROUND(*number,num_digits*)
Rounds a number to a specified number of digits

SIGN(*number*)
Returns the sign of a number

SQRT(*number*)
Returns the square root of a number

SUM(*number1,number2,...*)
Adds its arguments

TRUNC(*number,num_digits*)
Truncates a number to an integer

### Matrix

MDETERM(*array*)
Returns the matrix determinant of an array

MINVERSE(*array*)
Returns the matrix inverse of an array

MMULT(*array1,array2*)
Returns the matrix product of two arrays

SUMPRODUCT(*array1,array2,...*)
Multiplies corresponding components in the given arrays and returns the sum of those products

TRANSPOSE(*array*)
Returns the transpose of an array



### Statistical

AVERAGE(*number1,number2,...*)
Returns the average of its arguments

COUNT(*value1,value2,...*)
Counts how many numbers are in the list of arguments

COUNTA(*value1,value2,...*)
Counts how many values are in the list of arguments

GROWTH(*known_y's,known_x's,new_x's,const*)
Returns values along an exponential trend
$y = b*m\text{^}x$

LINEST(*known_y's,known_x's,const,stats*)
Returns the parameters of a linear trend
$y = m_1x_1 + m_2x_2 + \cdots + b$ or $y = m*x + b$

LOGEST(*known_y's,known_x's,const,stats*)
Returns the parameters of an exponential trend
$y = (b*(m_1\text{^}x_1)*(m_2\text{^}x_2)*\cdots)$ or $y = b*m\text{^}x$

MAX(*number1,number2,...*)
Returns the maximum value in a list of arguments

MEDIAN(*number1,number2,...*)
Returns the median of numbers

MIN(*number1,number2,...*)
Returns the minimum value in a list of arguments

STDEV(*number1,number2,...*)
Estimates standard deviation based on a sample

STDEVP(*number1,number2,...*)
Calculates standard deviation based on the entire population

TREND(*known_y's,known_x's,new_x's,const*)
Returns values along a linear trend
$y = m*x + b$

VAR(*number1,number2,...*)
Estimates variance based on a sample

VARP(*number1,number2,...*)
Calculates variance based on the entire population

### Text

CHAR(*number*)
Returns the character specified by the code number

CLEAN(*text*)
Removes control characters from text

CODE(*text*)
Returns the code number of the first character in text

DOLLAR(*number,decimals*)
Formats a number and converts it to text

EXACT(*text1,text2*)
Checks to see if two text values are identical

FIND(*find_text,within_text,start_at_num*)
Finds one text value within another (case sensitive)

FIXED(*number,decimals*)
Formats a number as text with a fixed number of decimals

LEFT(*text,num_chars*)
Extracts the leftmost characters from a text value

LEN(*text*)
Returns the length of a text string

LOWER(*text*)
Converts text to lowercase

MID(*text,start_num,num_chars*)
Extracts a number of characters from text

PROPER(*text*)
Converts text to initial capitals

REPLACE(*old_text,start_num,num_chars, new_text*)
Replaces characters within text

REPT(*text,number_times*)
Repeats text a given number of times

RIGHT(*text,num_chars*)
Extracts the rightmost characters from a text value

SEARCH(*find_text,within_text,start_num*)
Finds one text value within another (not case sensitive)

SUBSTITUTE(*text,old_text,new_text, instance_num*)
Replaces characters within text

TEXT(*value,format_text*)
Formats a number and converts it to text

TRIM(*text*)
Removes spaces from text

UPPER(*text*)
Capitalizes every letter in a text value

VALUE(*text*)
Converts a text argument to a number

### *Trigonometric*

ACOS(*number*)
Returns the arccosine of a number

ACOSH(*number*)
Returns the inverse hyperbolic cosine of a number

ASIN(*number*)
Returns the arcsine of a number

ASINH(*number*)
Returns the inverse hyperbolic sine of a number

ATAN(*number*)
Returns the arctangent of a number

ATAN2(*x_num,y_num*)
Returns the arctangent from *x*- and *y*-coordinates

ATANH(*number*)
Returns the inverse hyperbolic tangent of a number

COS(*number*)
Returns the cosine of a number

COSH(*number*)
Returns the hyperbolic cosine of a number

SIN(*number*)
Returns the sine of a number

SINH(*number*)
Returns the hyperbolic sine of a number

TAN(*number*)
Returns the tangent of a number

TANH(*number*)
Returns the hyperbolic tangent of a number

## *Macro Functions by Action*

### *Command-equivalent functions*

| Command | Function |
|---|---|
| Chart Add Arrow | ADD.ARROW |
| Chart Add Legend | LEGEND |
| Chart Add Overlay | ADD.OVERLAY |
| Chart Attach Text | ATTACH.TEXT |
| Chart Axes | AXES |
| Chart Calculate Now | CALCULATE.NOW |
| Chart Delete Arrow | DELETE.ARROW |
| Chart Delete Overlay | DELETE.OVERLAY |
| Chart Edit Series | EDIT.SERIES |
| Chart Gridlines | GRIDLINES |
| Chart Select Chart | SELECT.CHART |
| Chart Select Plot Area | SELECT.PLOT.AREA |
| Control Close | CLOSE |
| Control Maximize | APP.MAXIMIZE |
| Control Minimize | APP.MINIMIZE |
| Control Move (application) | APP.MOVE |
| Control Move (document) | MOVE |
| Control Restore | APP.RESTORE |
| Control Size (application) | APP.SIZE |
| Control Size (document) | SIZE |
| Data Consolidate | CONSOLIDATE |
| Data Delete | DATA.DELETE |
| Data Extract | EXTRACT |
| Data Find | DATA.FIND |
| Data Find | DATA.FIND.NEXT |
| Data Find | DATA.FIND.PREV |
| Data Form | DATA.FORM |
| Data Parse | PARSE |
| Data Series | DATA.SERIES |
| Data Set Criteria | SET.CRITERIA |
| Data Set Database | SET.DATABASE |
| Data Set Extract | SET.EXTRACT |
| Data Sort | SORT |
| Data Table | TABLE |
| Edit Clear | CLEAR |
| Edit Copy | COPY |
| Edit Copy Chart | COPY.CHART |



| Command | Function |
|---|---|
| Edit Copy Picture | COPY.PICTURE |
| Edit Create Publisher | CREATE.PUBLISHER |
| Edit Cut | CUT |
| Edit Delete | EDIT.DELETE |
| Edit Fill Down | FILL.DOWN |
| Edit Fill Left | FILL.LEFT |
| Edit Fill Right | FILL.RIGHT |
| Edit Fill Up | FILL.UP |
| Edit Fill Workgroup | FILL.WORKGROUP |
| Edit Insert | INSERT |
| Edit Paste | PASTE |
| Edit Paste Link | PASTE.LINK |
| Edit Paste Picture | PASTE.PICTURE |
| Edit Paste Picture Link | PASTE.PICTURE.LINK |
| Edit Paste Special | PASTE.SPECIAL |
| Edit Repeat | EDIT.REPEAT |
| Edit Subscribe To | SUBSCRIBE.TO |
| Edit Undo | UNDO |
| File Close | FILE.CLOSE |
| File Close All | CLOSE.ALL |
| File Delete | FILE.DELETE |
| File Exit | QUIT |
| File Links | CHANGE.LINK |
| File Links | LINKS |
| File Links | OPEN.LINKS |
| File Links | UPDATE.LINK |
| File New | NEW |
| File Open | OPEN |
| File Open Mail | OPEN.MAIL |
| File Page Setup | PAGE.SETUP |
| File Print | PRINT |
| File Print Preview | PRINT.PREVIEW |
| File Printer Setup | PRINTER.SETUP |
| File Quit | QUIT |
| File Save | SAVE |
| File Save As | SAVE.AS |
| File Save Workspace | SAVE.WORKSPACE |
| File Send Mail | SEND.MAIL |
| Format 3-D View | VIEW.3D |
| Format Alignment | ALIGNMENT |
| Format Border | BORDER |
| Format Bring To Front | BRING.TO.FRONT |
| Format Cell Protection | CELL.PROTECTION |
| Format Column Width | COLUMN.WIDTH |
| Format Font | FORMAT.FONT |
| Format Group | GROUP |
| Format Justify | JUSTIFY |
| Format Legend | FORMAT.LEGEND |
| Format Main Chart | FORMAT.MAIN |
| Format Move | FORMAT.MOVE |
| Format Number | FORMAT.NUMBER |
| Format Object Placement | PLACEMENT |
| Format Object Protection | OBJECT.PROTECTION |
| Format Overlay | FORMAT.OVERLAY |
| Format Patterns | PATTERNS |
| Format Row Height | ROW.HEIGHT |
| Format Scale | SCALE |
| Format Send To Back | SEND.TO.BACK |
| Format Size | FORMAT.SIZE |
| Format Style | APPLY.STYLE |
| Format Style | DEFINE.STYLE |
| Format Style | DELETE.STYLE |
| Format Style | MERGE.STYLES |

| Command | Function |
|---|---|
| Format Text | FORMAT.TEXT |
| Format Ungroup | UNGROUP |
| Formula Apply Names | APPLY.NAMES |
| Formula Create Names | CREATE.NAMES |
| Formula Define Name | DEFINE.NAME |
| Formula Define Name | DELETE.NAME |
| Formula Find | FORMULA.FIND |
| Formula Find | FORMULA.FIND.NEXT |
| Formula Find | FORMULA.FIND.PREV |
| Formula Goal Seek | GOAL.SEEK |
| Formula Goto | FORMULA.GOTO |
| Formula Note | NOTE |
| Formula Outline | OUTLINE |
| Formula Paste Name | LIST.NAMES |
| Formula Replace | FORMULA.REPLACE |
| Formula Select Special | SELECT.LAST.CELL |
| Formula Select Special | SELECT.SPECIAL |
| Formula Show Active Cell | SHOW.ACTIVE.CELL |
| Gallery 3-D Area | GALLERY.3D.AREA |
| Gallery 3-D Column | GALLERY.3D.COLUMN |
| Gallery 3-D Line | GALLERY.3D.LINE |
| Gallery 3-D Pie | GALLERY.3D.PIE |
| Gallery Area | GALLERY.AREA |
| Gallery Bar | GALLERY.BAR |
| Gallery Column | GALLERY.COLUMN |
| Gallery Combination | COMBINATION |
| Gallery Line | GALLERY.LINE |
| Gallery Pie | GALLERY.PIE |
| Gallery Preferred | PREFERRED |
| Gallery Set Preferred | SET.PREFERRED |
| Gallery XY (Scatter) | GALLERY.SCATTER |
| Macro Assign To Object | ASSIGN.TO.OBJECT |
| Macro Run | RUN |
| Options Calculate Document | CALCULATE.DOCUMENT |
| Options Calculate Now | CALCULATE.NOW |
| Options Calculation | CALCULATION |
| Options Calculation | PRECISION |
| Options Color Palette | COLOR.PALETTE |
| Options Color Palette | EDIT.COLOR |
| Options Display | DISPLAY |
| Options Freeze Panes | FREEZE.PANES |
| Options Full Menus/ Short Menus | SHORT.MENUS |
| Options Protect Document | PROTECT.DOCUMENT |
| Options Remove Page Break | REMOVE.PAGE.BREAK |
| Options Set Page Break | SET.PAGE.BREAK |
| Options Set Print Area | SET.PRINT.AREA |
| Options Set Print Titles | SET.PRINT.TITLES |
| Options Standard Font | STANDARD.FONT |
| Options Workspace | WORKSPACE |
| Window Arrange All | ARRANGE.ALL |
| Window Hide | HIDE |
| Window New Window | NEW.WINDOW |
| Window Show Clipboard | SHOW.CLIPBOARD |
| Window Show Info | SHOW.INFO |
| Window Unhide | UNHIDE |
| Window Workgroup | WORKGROUP |



### *Other action-equivalent macro functions*

| Function | Action |
|---|---|
| A1.R1C1 | Displays A1 or R1C1 references |
| ACTIVATE | Activates a window |
| ACTIVATE.NEXT | Activates the next window |
| ACTIVATE.PREV | Activates the previous window |
| APP.ACTIVATE | Activates another application |
| BEEP | Sounds a tone |
| CANCEL.COPY | Cancels the copy marquee |
| CREATE.OBJECT | Creates an object |
| DELETE.FORMAT | Deletes a number format |
| DEMOTE | Demotes the selection |
| DUPLICATE | Makes a copy of an object |
| EDITION.OPTIONS | Sets options for a publisher or subscriber |
| FORMULA | Enters values into a cell or range |
| FORMULA.ARRAY | Enters an array |
| FORMULA.CONVERT | Changes the reference style and type |
| FORMULA.FILL | Fills the selection |
| FULL | Changes the size of the active window |
| HIDE.OBJECT | Hides an object |
| HLINE | Horizontally scrolls the active window by columns |
| HPAGE | Horizontally scrolls the active window one window at a time |
| HSCROLL | Horizontally scrolls a document by percentage or by column number |
| PROMOTE | Promotes the selection |
| SELECT | Selects a cell, a worksheet object, or a chart item |
| SELECT.END | Selects the last cell in a range |
| SET.UPDATE.STATUS | Controls the update status of a link |
| SHOW.DETAIL | Expands or collapses the detail under an outline summary icon |
| SHOW.LEVELS | Shows outline levels |
| SPLIT | Splits a window |
| TEXT.BOX | Replaces text in a text box |
| UNLOCKED.NEXT | Goes to the next unlocked cell |
| UNLOCKED.PREV | Goes to the previous unlocked cell |
| VLINE | Vertically scrolls the active window by rows |
| VPAGE | Vertically scrolls the active window one window at a time |
| VSCROLL | Vertically scrolls a document by percentage or row number |

### *Customizing macro functions*

| Function | Action |
|---|---|
| ADD.BAR | Adds a menu bar |
| ADD.COMMAND | Adds a command |
| ADD.MENU | Adds a menu |

| Function | Action |
|---|---|
| ALERT | Displays a dialog box and a message |
| CALL | Calls a procedure in a code resource |
| CANCEL.KEY | Disables macro interruption |
| CHECK.COMMAND | Marks a command |
| CUSTOM.REPEAT | Specifies a macro to run when Edit Repeat is chosen |
| CUSTOM.UNDO | Specifies a macro to run when Edit Undo is chosen |
| DELETE.BAR | Deletes a menu bar |
| DELETE.COMMAND | Deletes a command |
| DELETE.MENU | Deletes a menu |
| DIALOG.BOX | Displays a custom dialog box |
| DISABLE.INPUT | Blocks all input to Microsoft Excel |
| ECHO | Controls screen updating |
| ENABLE.COMMAND | Enables or disables a menu or custom command |
| GET.BAR | Returns information about the active menu bar |
| HELP | Displays a custom Help topic |
| MESSAGE | Displays a message in the status bar |
| RENAME.COMMAND | Changes the name of a command or menu |
| SHOW.BAR | Displays a menu bar |

### *Control macro functions*

| Function | Action |
|---|---|
| ARGUMENT | Passes an argument to a macro |
| BREAK | Interrupts a FOR, a FOR.CELL, or a WHILE loop |
| ELSE | Specifies an action to take if an IF function returns FALSE |
| ELSE.IF | Specifies an action to take if an IF or another ELSE.IF function returns FALSE |
| END.IF | Ends a group of macro functions started with an IF statement |
| ERROR | Turns error checking on and off |
| EXEC | Starts another application |
| EXECUTE | Carries out a command in another application |
| FCLOSE | Closes a text file |
| FOPEN | Opens a text file |
| FOR | Starts a FOR–NEXT loop |
| FOR.CELL | Starts a FOR.CELL–NEXT loop through a reference |
| FPOS | Sets the position in a text file |
| FREAD | Reads characters from a text file |
| FREADLN | Reads a line from a text file |
| FSIZE | Returns the size of a text file |
| FWRITE | Writes characters to a text file |
| FWRITELN | Writes a line to a text file |



| Function | Action |
|---|---|
| GOTO | Directs macro execution to another cell |
| HALT | Stops all macros from running |
| IF | Specifies an action to take if a logical test is TRUE |
| INITIATE | Opens a channel to another application |
| INPUT | Displays a simple dialog box |
| NEXT | Ends a WHILE, FOR, or FOR.CELL loop |
| ON.DATA | Runs a macro when data is sent to Microsoft Excel by another application |
| ON.KEY | Runs a macro when a key is pressed |
| ON.RECALC | Runs a macro when a document is recalculated |
| ON.TIME | Runs a macro at a specific time |
| ON.WINDOW | Runs a macro when a window is activated |
| POKE | Sends data to another application |
| REGISTER | Registers a code resource |
| REQUEST | Returns data from another application |
| RESTART | Removes return addresses from the stack |
| RESULT | Specifies the data type a custom function returns |
| RETURN | Ends the currently running macro |
| SEND.KEYS | Sends a key sequence to an application |
| SET.NAME | Defines a name as a value |
| SET.VALUE | Enters values in a cell |
| STEP | Turns on macro single-stepping |
| TERMINATE | Closes a channel to another application |
| UNREGISTER | Removes a registered code resource from memory |
| VOLATILE | Makes custom functions recalculate automatically |
| WAIT | Pauses a macro |
| WHILE | Starts a WHILE–NEXT loop |

### *Value-returning functions*

| Function | Action |
|---|---|
| ABSREF | Returns the reference of a range that has a relative relationship to another range |
| ACTIVE.CELL | Returns the reference of the active cell |
| CALLER | Returns the reference of the calling function |
| DEREF | Returns the value of the cells in the reference |
| DIRECTORY | Sets the current drive and directory to a specified path |
| DOCUMENTS | Returns the names of all open documents |
| FILES | Returns the filenames in the specified directory or folder |
| GET.CELL | Returns information about the specified cell |

| Function | Action |
|---|---|
| GET.CHART.ITEM | Returns information about a chart item |
| GET.DEF | Returns a name matching a definition |
| GET.DOCUMENT | Returns information about a document |
| GET.FORMULA | Returns the contents of a cell |
| GET.LINK.INFO | Returns information about a link |
| GET.NAME | Returns the definition of a name |
| GET.NOTE | Returns characters from a note |
| GET.OBJECT | Returns information about an object |
| GET.WINDOW | Returns information about a window |
| GET.WORKSPACE | Returns information about the workspace |
| LAST.ERROR | Returns the reference of the cell where the last error occurred |
| NAMES | Returns the names defined in a document |
| REFTEXT | Converts a reference to text |
| RELREF | Returns a relative reference |
| SELECTION | Returns the reference of the selection |
| TEXTREF | Converts text to a reference |
| WINDOWS | Returns the names of all open windows |

## Differences Due to Operating Environment

Most of the information in this book applies to all users of Microsoft Excel. However, there are some functions that are not supported by both Microsoft Excel for the Macintosh and Microsoft Excel for Windows or OS/2. You can create macros for use in another operating environment using macro functions that are not supported by your current operating environment. The macro functions will be accepted by the macro sheet when you type them but will return error values when you try to run them in the unsupported environment.

The following functions in Microsoft Excel for Windows and OS/2 are not supported by Microsoft Excel for the Macintosh:

- APP.MAXIMIZE
- APP.MINIMIZE
- APP.MOVE
- APP.RESTORE
- APP.SIZE
- PRINTER.SETUP
- SEND.KEYS

The following functions in Microsoft Excel for the Macintosh are not supported by Microsoft Excel for Windows or OS/2:

- OPEN.MAIL
- SEND.MAIL

The following functions are only supported by Microsoft Excel for the Macintosh if you are using system software version 7.0 or later:

- APP.ACTIVATE
- EXEC
- EXECUTE
- INITIATE
- ON.DATA
- POKE
- REQUEST
- TERMINATE



The following functions are only supported by Microsoft Excel for the Macintosh if you are using system software version 7.0 or later. They are not supported by Microsoft Excel for Windows or OS/2:

- CREATE.PUBLISHER
- SUBSCRIBE.TO

The following functions are only supported by Microsoft Excel for Windows if you are using Hewlett Packard® NewWave™:

- ATTRIBUTES
- SAVE.NEW.OBJECT
- SHARE
- SHARE.NAME

# Function Reference

*Macro Sheets Only*

## A1.R1C1(logical)

Displays row and column headings and cell references in either the R1C1 or A1 reference style. A1 is the Microsoft Excel default reference style.

***Logical*** is a logical value specifying which reference style to use. If *logical* is TRUE, all worksheets and macro sheets use A1 references; if FALSE, all worksheets and macro sheets use R1C1 references.

### Example

The following macro formula displays an alert box asking you to choose either A1 or R1C1 reference style. This is useful in an Auto_Open macro if several people who prefer different reference styles must maintain the same worksheet.

```
A1.R1C1(ALERT("Click OK for A1 style; Cancel for R1C1",1))
```

## ABS(number)

Returns the absolute value of a number. The absolute value of a number is the number without its sign.

***Number*** is the real number of which you want the absolute value.

### Examples

ABS(3) equals 3

ABS(-2) equals 2

If A1 contains –16, then:
SQRT(ABS(A1)) equals 4

### Related Function

SIGN Returns the sign of a number as a value: 1 (positive), –1 (negative), or 0 (zero)

*Macro Sheets Only*

## ABSREF(ref_text,reference)

Returns the absolute reference of the cells that are offset from a reference by a specified amount. You should generally use OFFSET instead of ABSREF. This function is provided for users who prefer to supply a relative reference in the form of text as an argument.

***Ref_text*** specifies a position relative to *reference*. Think of *ref_text* as "directions" from one range of cells to another.

- *Ref_text* must be an R1C1-style relative reference in the form of text, such as "R[1]C[1]".
- *Ref_text* is considered relative to the cell in the upper-left corner of *reference*.

***Reference*** is a cell or range of cells specifying a "starting point" that *ref_text* uses to locate another range of cells. *Reference* can be an external reference.

### Notes

- If you use ABSREF in a function or operation, you will usually get the values contained in the reference instead of the reference itself because the reference is automatically converted to the contents of the reference.
- If you use ABSREF in a function that requires a reference argument, then Microsoft Excel does not convert the reference to a value.
- If you want to work with the actual reference, use the REFTEXT function to convert the active-cell reference to text, which you can then store or manipulate (or convert back to a reference with TEXTREF). See the third example below.

### Examples

ABSREF("R[-2]C[2]",C3) equals $A$1

ABSREF(RELREF(A1,C3),D4) equals $B$2

REFTEXT(ABSREF("R[-2]C[-2]:R[2]C[2]",C3:G7),TRUE) equals REFTEXT(ABSREF("R[-2]C[-2]:R[2]C[2]",C3),TRUE) equals "$A$1:$E$5"

In Microsoft Excel for Windows or OS/2, if the document in the active window is named FINANCE.XLS:

ABSREF("R[-2]C[-2]",FINANCE.XLS!C3) equals FINANCE.XLS!$A$1

In Microsoft Excel for the Macintosh, if the document in the active window is named Finance:

ABSREF("R[-2]C[-2]",Finance!C3) equals Finance!$A$1

### Related Functions

| | |
|---|---|
| OFFSET | Creates a reference offset from the active selection |
| RELREF | Returns a relative reference |

## ACOS(number)

Returns the arccosine of a number. The arccosine is the angle whose cosine is *number*. The returned angle is given in radians in the range 0 to π.

***Number*** is the cosine of the angle you want and must be from –1 to 1.

If you want to convert the result from radians to degrees, multiply it by 180/PI().

### Examples

ACOS(-0.5) equals 2.094 (2π/3 radians)

ACOS(-0.5)*180/PI() equals 120 (degrees)

### Related Functions

| | |
|---|---|
| COS | Returns the cosine of a number |
| PI | Returns the value π |

## ACOSH(number)

Returns the inverse hyperbolic cosine of *number*. *Number* must be greater than or equal to 1. The inverse hyperbolic cosine is the value whose hyperbolic cosine is *number*, so ACOSH(COSH(*number*)) equals *number*.

### Examples

ACOSH(1) equals 0

ACOSH(10) equals 2.993222846

### Related Functions

| | |
|---|---|
| ASINH | Returns the inverse hyperbolic sine of a number |
| ATANH | Returns the inverse hyperbolic tangent of a number |
| COSH | Returns the hyperbolic cosine of a number |

*Macro Sheets Only*

## ACTIVATE(window_text,pane_num)

Activates a window if more than one window is open, or a pane of a window if the window is split. Activating a pane is useful with functions such as VSCROLL, HSCROLL, and GOTO, which operate only on the active pane.

***Window_text*** is the name of a window to activate in the form of text, for example, "Sheet1" or "Sheet1:2".

- If a document is displayed in more than one window and *window_text* does not specify which window to activate, the first window containing that document is activated.
- If *window_text* is omitted, the active window is not changed.

***Pane_num*** is a number from 1 to 4 specifying which pane to activate. If *pane_num* is omitted and the window has more than one pane, the active pane is not changed when *window_text* is activated.

| *Pane_num* | Pane that is activated |
|---|---|
| 1 | Top left. If the window is not split, this is the only pane. If the window is split only horizontally, this is the top pane. If the window is split only vertically, this is the left pane. |
| 2 | Top right. If the window is split only vertically, this is the right pane. If the window is split only horizontally, an error occurs. |
| 3 | Bottom left. If the window is split only horizontally, this is the bottom pane. If the window is split only vertically, an error occurs. |
| 4 | Bottom right. If the window is split into only two panes, an error occurs. |



### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula activates the top pane of a horizontally split window on a macro sheet called MACRO1.XLM:

ACTIVATE("MACRO1.XLM",1)

The following macro formula activates a chart called CHART3.XLC:

ACTIVATE("CHART3.XLC")

In Microsoft Excel for the Macintosh, the following macro formula activates the top pane of a horizontally split window on a macro sheet called Macro1:

ACTIVATE("Macro1",1)

The following macro formula activates a chart called SalesChart:

ACTIVATE("SalesChart")

Also see the example for NAMES.

### Related Functions

| | |
|---|---|
| ACTIVATE.NEXT and ACTIVATE.PREV | Activate the next or previous window, respectively |
| FREEZE.PANES | Freezes the panes of a window so that they do not scroll |
| ON.WINDOW | Runs a macro when a specified window is activated |
| SPLIT | Splits the active window into panes |

*Macro Sheets Only*

## ACTIVATE.NEXT()
## ACTIVATE.PREV()

Activates the next or previous window, respectively. These functions are equivalent to pressing CTRL+F6 or CTRL+SHIFT+F6 in Microsoft Excel for Windows or OS/2, or COMMAND+M or COMMAND+SHIFT+M in Microsoft Excel for the Macintosh.

### Related Functions

| | |
|---|---|
| ACTIVATE | Activates a window or pane |
| ON.WINDOW | Runs a macro when a specified window is activated |

*Macro Sheets Only*

## ACTIVE.CELL()

Returns the reference of the active cell in the selection as an external reference. If an object is selected or a chart is active, ACTIVE.CELL returns the #N/A error value.

### Notes

- If you use ACTIVE.CELL in a function or operation, you will usually get the value contained in the active cell instead of its reference because the reference is automatically converted to the contents of the reference. See the third example below.
- If you use ACTIVE.CELL in a function that requires a reference argument, then Microsoft Excel does not convert the reference to a value.
- If you want to work with the actual reference, use the REFTEXT function to convert the active-cell reference to text, which you can then store or manipulate (or convert back to a reference with TEXTREF). See the second example below.

### Examples

The following macro formula assigns the name Sales to the active cell:

SET.NAME("Sales",ACTIVE.CELL())

In the previous example, note that "Sales" refers to a cell on the active worksheet, but the name itself exists only in the macro sheet's list of names. In other words, the preceding formula does not define a name on the worksheet.

The following macro formula puts the reference of the active cell into the cell named Temp:

```
FORMULA("="&REFTEXT(ACTIVE.CELL()),"Temp")
```

The following macro formula checks the contents of the active cell. If the cell contains only the letter "c" or "s", the macro branches to an area named FinishRefresh:

```
IF(OR(ACTIVE.CELL()="c",ACTIVE.CELL()="s"),
GOTO(FinishRefresh))
```

In Microsoft Excel for Windows or OS/2, if the document in the active window is named SALES.XLS and A1 is the active cell, then:

```
ACTIVE.CELL() equals SALES.XLS!$A$1
```

In Microsoft Excel for the Macintosh, if the document in the active window is named Sales 1 and A1 is the active cell, then:

```
ACTIVE.CELL() equals 'Sales 1'!$A$1
```

To see ACTIVE.CELL working with other functions, see the example for PARSE.

### Related Functions

| | |
|---|---|
| OFFSET | Returns a reference offset from a given reference |
| SELECT | Selects cells on a worksheet or an item on a chart |

Macro Sheets Only

## ADD.ARROW()

Equivalent to the Chart Add Arrow command. Adds an arrow to the active chart. If a chart is not the active document, displays an error value. For detailed information on the Chart Add Arrow command, see online Help.

After you create an arrow with ADD.ARROW, the arrow remains selected, so you can use the arrow form of the PATTERNS function to format the arrow and the FORMAT.MOVE and FORMAT.SIZE functions to change the position and size of the arrow.

### Related Functions

| | |
|---|---|
| DELETE.ARROW | Deletes the selected arrow |
| FORMAT.MOVE | Moves the selected object |
| FORMAT.SIZE | Changes the size of the selected object |
| PATTERNS | Changes the appearance of the selected object |

Macro Sheets Only

## ADD.BAR(*bar_num*)

Creates a new menu bar and returns the bar ID number. Use the bar ID number to identify the menu in functions that display and add menus and commands to the menu bar. You can also use ADD.BAR to restore a built-in menu bar with its original menus and commands.

***Bar_num*** is the number of a built-in menu bar that you want to restore. Use ADD.BAR(*bar_num*) to restore an unaltered version of a built-in menu bar after you have made changes to the menu bar's menus and commands. See ADD.COMMAND for a list of ID numbers for built-in menu bars.

### Important

Restoring a built-in menu bar will remove menus and commands added by other macros. Use ADD.COMMAND and ADD.MENU to restore individual commands and menus.

### Note

ADD.BAR just creates a new menu bar; it does not display it. Use SHOW.BAR to display a menu bar. The argument to the SHOW.BAR function should be the number returned by ADD.BAR or a reference to the cell containing ADD.BAR.



You can define up to 15 custom menu bars at one time, in addition to Microsoft Excel's six built-in menu bars. If you carry out an ADD.BAR function when more than 15 custom menu bars are already defined, Microsoft Excel returns the #VALUE! error value.

For information on custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Example

The following formula creates a new menu bar and returns a bar ID number:

```
ADD.BAR()
```

### Related Functions

| | |
|---|---|
| ADD.COMMAND | Adds a command to a menu |
| ADD.MENU | Adds a menu to a menu bar |
| DELETE.BAR | Deletes a menu bar |
| SHOW.BAR | Displays a menu bar |

Macro Sheets Only

## ADD.COMMAND(*bar_num, menu, command_ref, position*)

Adds a command to a menu. Use ADD.COMMAND to add one or more custom menu commands to a menu on a built-in or custom menu bar. ADD.COMMAND returns the position number on the menu of the added command. You can also use ADD.COMMAND to restore a deleted built-in command to its original menu.

***Bar_num*** is the menu bar to which you want a command added. *Bar_num* can be the ID number of a built-in or custom menu bar. The ID number of a custom menu bar is the number returned by the ADD.BAR function.

The ID numbers of the built-in menu bars are:

| Number | Built-in menu bar |
|---|---|
| 1 | Worksheet and macro sheet menu bar (full menus) |
| 2 | Chart menu bar (full menus) |
| 3 | Null menu bar (the menu displayed when no documents are open) |
| 4 | Info menu bar |
| 5 | Worksheet and macro sheet menu bar (short menus) |
| 6 | Chart menu bar (short menus) |

***Menu*** is the menu to which you want the new command added. *Menu* can be either the name of a menu as text or the number of a menu. Menus are numbered starting with 1 from the left of the screen.

***Command_ref*** is a reference to an area on the macro sheet that describes the new command or commands. *Command_ref* must be at least two columns wide. You can use the name of a built-in command that has been deleted to restore that command on its original menu. You can also use a reference to the cell containing the DELETE.COMMAND formula that deleted the command you want to restore. *Command_ref* is similar to *menu_ref* in ADD.MENU. For more information about *command_ref*, see the description of *menu_ref* in ADD.MENU.

***Position*** specifies the placement of the new command. *Position* can be the name of a command, as quoted text, or the number of a command. Commands are numbered from the top of the menu starting with 1. Commands are added above the position specified. If *position* is omitted, the command is added to the bottom of the menu.

### Tip

Always use menu and command names rather than numbers for arguments. The numbers assigned to menus and commands change as you add and delete menus and commands. Using names ensures that your menu and command macro functions always refer to the items you intend.

### Note

Later ADD.COMMAND functions that do not include a *position* argument will occupy lower lines in the menu.

For more information about using custom menu bars, menus, and commands, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Examples

The following macro formulas add the command described in cells G16:J16 to the Formula menu (full menus and short menus):

```
ADD.COMMAND(1,"Formula",G16:J16)
ADD.COMMAND(5,"Formula",G16:J16)
```

Also see the example in SHOW.BAR.

### Related Functions

| | |
|---|---|
| ADD.BAR | Creates a new menu bar |
| ADD.MENU | Adds a menu to a menu bar |
| CHECK.COMMAND | Adds or deletes a check mark from a command |
| DELETE.COMMAND | Deletes a command from a menu |
| ENABLE.COMMAND | Enables or disables a command |
| RENAME.COMMAND | Renames a command |

Macro Sheets Only

## ADD.MENU(*bar_num,menu_ref,position*)

Adds a menu to a menu bar. Use ADD.MENU to add a custom menu to a built-in or custom menu bar. You can also use ADD.MENU to restore built-in menus you have deleted with DELETE.MENU. ADD.MENU returns the position number in the menu bar of the new menu.

***Bar_num*** is the menu bar to which you want a menu added. *Bar_num* can be the ID number of a built-in or custom menu bar. See ADD.COMMAND for a list of ID numbers for built-in menu bars.

***Menu_ref*** is a reference to an area on the macro sheet that describes the new menu or the name of a deleted built-in menu you want to restore.

- *Menu_ref* must be made up of at least two rows and two columns of cells. The upper-left cell of *menu_ref* specifies the menu title, which is displayed in the menu bar. In the following example, the range A3:E10 is a valid *menu_ref*.

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Menu or command name | Macro name | Shortcut key | Status bar | Help topics |
| 2 | | | *Macintosh only* | | |
| 3 | Reports | | | | |
| 4 | Weekly Report | WeeklyRept | | Prints weekly report | Help!35 |
| 5 | Monthly Report | MonthlyRept | | Prints monthly report | Help!36 |
| 6 | Quarterly Report | QuartRept | | Prints quarterly report | Help!37 |
| 7 | . | | | | |
| 8 | Custom Report | CustomRpt | | Create a custom report | Help!38 |
| 9 | . | | | | |
| 10 | Remove Menu | RemoveMen | | Removes Reports menu | Help!39 |

- The rest of the first column indicates the names of the commands. The corresponding rows in the second column give the names of the macros that run when the commands are chosen.
- You can also specify status-bar text and Help topics in *menu_ref*. In Microsoft Excel for the Macintosh, you can specify shortcut keys in *menu_ref*. For more information on creating custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

***Position*** specifies the placement of the new menu. *Position* can be the name of a menu, as quoted text, or the number of a menu. Menus are numbered from left to right starting with 1. Menus are added to the left of the position specified.

- If *position* is omitted, the menu is added to the end of the menu bar.



- If there is already a menu at *position*, that menu is shifted to the right and the new menu is added in its place.
- If you are using ADD.MENU to restore a deleted built-in menu, you can use the *position* argument to put it back in its original place on the menu bar. For example, to restore the Data menu on the worksheet and macro sheet menu bar, use *position* 5. If *position* is omitted, the menu is added to the right of the last menu restored.

For more information on custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Example

The following formula adds a new menu to the worksheet menu bar (full menus), where A10:B15 is the *menu_ref* describing the menu:

```
ADD.MENU(1,A10:B15)
```

Also see the example in SHOW.BAR.

### Related Functions

| | |
|---|---|
| ADD.COMMAND | Adds a command to a menu |
| DELETE.MENU | Deletes a menu |
| ENABLE.COMMAND | Enables or disables a command |

Macro Sheets Only

## ADD.OVERLAY()

Equivalent to the Chart Add Overlay command (full menus). Adds an overlay to a 2-D chart. If the active chart already has an overlay, ADD.OVERLAY takes no action and returns TRUE. For detailed information on the Chart Add Overlay command, see online Help.

### Related Functions

| | |
|---|---|
| ADD.ARROW | Adds an arrow to a chart |
| LEGEND | Adds a legend to a chart |

## ADDRESS(*row_num,column_num,abs_num,a1,sheet_text*)

Creates a cell address as text, given specified row and column numbers. Use ADDRESS to create cell references as text.

***Row_num*** is the row number to use in the cell reference.

***Column_num*** is the column number to use in the cell reference.

***Abs_num*** specifies the type of reference to return, according to the following table. The default is 1, absolute reference.

| *Abs_num* | Specifies this type of reference |
|---|---|
| 1 | Absolute reference |
| 2 | Absolute row; relative column |
| 3 | Relative row; absolute column |
| 4 | Relative reference |

***A1*** is a logical value that specifies the A1 or R1C1 reference style. If *a1* is TRUE or omitted, ADDRESS returns an A1-style reference; if FALSE, ADDRESS returns an R1C1-style reference.

***Sheet_text*** is text specifying the name of the worksheet or macro sheet to be used as the external reference. If *sheet_text* is omitted, no sheet name is used.

### Examples

ADDRESS(2,3) equals $C$2

ADDRESS(2,3,2) equals C$2

ADDRESS(2,3,2,FALSE) equals R2C[3]

ADDRESS(2,3,2,FALSE,"SHEET1.XLS") equals SHEET1.XLS!C$2

ADDRESS(2,3,2,FALSE,"EXCEL SHEET") equals 'EXCEL SHEET'!C$2

### Related Functions

| | |
|---|---|
| ACTIVE.CELL | Returns the reference of the active cell |
| COLUMN | Returns the column number of a reference |
| OFFSET | Returns a reference offset from a given reference |
| ROW | Returns the row number of a reference |
| SELECTION | Returns the reference of the current selection |

*Macro Sheets Only*

## ALERT(*message_text,type_num*)

Displays a dialog box and message, and waits for you to choose a button. Use ALERT instead of MESSAGE if you want to interrupt the flow of a macro and force the user to make a choice or to notice an important message.

***Message_text*** is the message, enclosed in quotation marks, displayed in the dialog box.

***Type_num*** is a number from 1 to 3 specifying which type of dialog box to display. If you omit *type_num*, it is assumed to be 2.

- If *type_num* is 1, ALERT displays a dialog box with OK and Cancel buttons. Choose a button to continue or cancel an action. ALERT returns TRUE if you choose the OK button and FALSE if you choose the Cancel button. See the last example below.
- If *type_num* is 2 or 3, ALERT displays a dialog box with an OK button. Choose the button to continue, and ALERT returns TRUE. The only difference between specifying 2 or 3 is that ALERT displays a different icon on the left side of the dialog box. So, for example, you could use 2 for notes or to present general information, and 3 for errors or warnings.

### Note

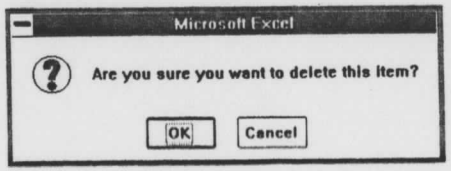
In Microsoft Excel for the Macintosh, the ALERT dialog box is not a moveable window.

### Examples

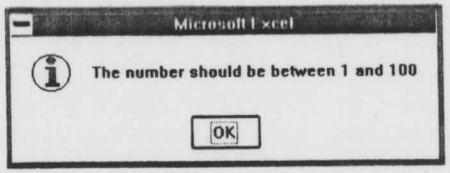
The following dialog boxes show the results of using ALERT with *type_num* 1, 2, and 3.

In Microsoft Excel for Windows or OS/2, the following macro formulas display these three dialog boxes:

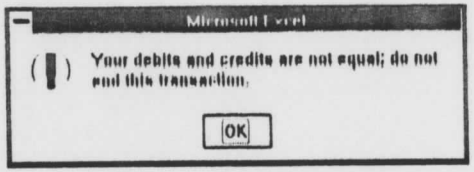
ALERT("Are you sure you want to delete this item?",1)




ALERT("The number should be between 1 and 100",2)

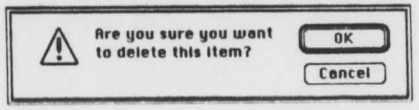



ALERT("Your debits and credits are not equal; do not end this transaction.",3)



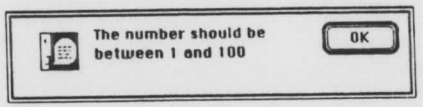

In Microsoft Excel for the Macintosh, the following macro formulas display these three dialog boxes:

ALERT("Are you sure you want to delete this item?",1)

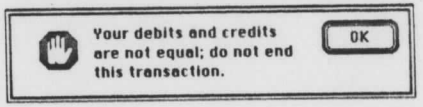





ALERT("The number should be between 1 and 100",2)




ALERT("Your debits and credits are not equal; do not end this transaction.",3)




A common use of the ALERT function is to give the user a choice of two actions. The following formula in an Auto_Open macro asks which reference style to use when the worksheet is opened.

A1.R1C1(ALERT("Click OK for A1 style, Cancel for R1C1",1))

Also see the examples for ARGUMENT, SHOW.CLIPBOARD, and the macro-only form of the IF function.

### Related Functions

| | |
|---|---|
| INPUT | Displays a dialog box for user input |
| MESSAGE | Displays a message in the status bar |

*Macro Sheets Only*

## ALIGNMENT(*type_num,wrap*)
## ALIGNMENT?(*type_num,wrap*)

Equivalent to the Format Alignment command. Aligns the contents of the selected cells. For detailed information on the Format Alignment command, see online Help.

***Type_num*** is a number from 1 to 5 specifying the type of alignment:

| *Type_num* | Alignment |
|---|---|
| 1 | General |
| 2 | Left |
| 3 | Center |
| 4 | Right |
| 5 | Fill |

***Wrap*** is a logical value corresponding to the Wrap Text check box in the Format Alignment dialog box. If *wrap* is TRUE, Microsoft Excel turns on the check box and text wraps in cells; if FALSE, Microsoft Excel turns off the check box and text does not wrap. The default is FALSE.

## AND(*logical1,logical2,...*)

Returns TRUE if all the arguments are TRUE; returns FALSE if one or more arguments is FALSE.

***Logical1, logical2,...*** are 1 to 14 conditions you want to test that can be either TRUE or FALSE.

- The arguments should be logical values, or arrays or references that contain logical values.
- If an array or reference argument contains text or empty cells, those values are ignored.
- If the specified range contains no logical values, AND returns the #VALUE! error value.

### Examples

AND(TRUE,TRUE) equals TRUE:

AND(TRUE,FALSE) equals FALSE

AND(2+2=4,2+3=5) equals TRUE

If B1:B3 contains the values TRUE, FALSE, and TRUE, then:

AND(B1:B3) equals FALSE

If B4 contains a number between 1 and 100, then:

AND(1<B4,B4<100) equals TRUE

Suppose you want to display B4 if it contains a number strictly between 1 and 100, and you want to display a message if it is not. If B4 contains 104, then:

IF(AND(1<B4,B4<100),B4,"The value is out of range.") equals "The value is out of range."

If B4 contains 50, then:

IF(AND(1<B4,B4<100),B4,"The value is out of range.") equals 50

### Related Functions

| | |
|---|---|
| NOT | Reverses the logic of its argument |
| OR | Returns TRUE if any argument is TRUE |

Macro Sheets Only

## APP.ACTIVATE(*title_text,wait_logical*)

Activates an application. Use APP.ACTIVATE to activate another application that is already running or that you have started by using EXEC.

### Important

Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

***Title_text*** is the name of an application as displayed in its title bar.

- If *title_text* is omitted, APP.ACTIVATE activates Microsoft Excel.
- If *title_text* is not a currently running application, APP.ACTIVATE returns the #VALUE! error value and interrupts the macro.
- *Title_text* is not necessarily the name of the application file. Use the text that appears in the title bar of the application, which might include the name of the open document and path information.
- In Microsoft Excel for the Macintosh, *title_text* can also refer to the Process Serial Number (PSN) that is returned by an EXEC function.

***Wait_logical*** is a logical value determining when the application specified by *title_text* is activated.

- If *wait_logical* is TRUE, Microsoft Excel waits to be activated before activating the application specified by *title_text*.
- If *wait_logical* is FALSE or omitted, Microsoft Excel immediately activates the application specified by *title_text*.

### Note

If you are running an application using Microsoft Excel macros, and you want to activate a third application without activating Microsoft Excel first, use FALSE as the *wait_logical* argument. This allows you to use the application *title_text* without having to activate Microsoft Excel.

### Examples

In Microsoft Excel for Windows or OS/2, this formula activates the Control Panel application as soon as Microsoft Excel is activated:

```
APP.ACTIVATE("Control Panel",TRUE)
```

The following macro formula activates Microsoft® Word, which is currently displaying the document MONTHRPT.DOC:

```
APP.ACTIVATE("Microsoft Word - MONTHRPT.DOC")
```

In Microsoft Excel for the Macintosh, the following macro formula activates Microsoft Word:

```
APP.ACTIVATE("Microsoft Word")
```

### Tip

Use an IF statement with APP.ACTIVATE to run an EXEC function if the application you want to activate is not yet running. For example, in Microsoft Excel for OS/2, the following macro runs an EXEC function that starts Control Panel if the APP.ACTIVATE function returns an error:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | ActivateControl | |
| 4 | | =ERROR(FALSE) | Turns off error checking |
| 5 | Act | =APP.ACTIVATE("Control Panel") | Activates the Control Panel |
| 6 | | =IF(ISERROR(Act),EXEC("PMCPL.EXE")) | If error, starts Control Panel |
| 7 | | =RETURN() | |



### Related Functions

The first five functions below are only for Microsoft Excel for Windows or OS/2.

| | |
|---|---|
| APP.MAXIMIZE | Maximizes the Microsoft Excel application window |
| APP.MINIMIZE | Minimizes the Microsoft Excel application window |
| APP.MOVE | Moves the Microsoft Excel application window |
| APP.RESTORE | Restores the Microsoft Excel application window |
| APP.SIZE | Changes the size of the Microsoft Excel application window |
| EXEC | Starts another application |

Macro Sheets Only

## APP.MAXIMIZE()

Equivalent to the Control Maximize command for the application window. Maximizes the Microsoft Excel window. For detailed information on the Control Maximize command, see online Help.

### Note

This function is only for Microsoft Excel for Windows or OS/2. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return the #N/A error value.

### Related Functions

| | |
|---|---|
| APP.ACTIVATE | Activates an application |
| APP.MINIMIZE | Minimizes the Microsoft Excel application window |
| APP.MOVE | Moves the Microsoft Excel application window |
| APP.RESTORE | Restores the Microsoft Excel application window |
| APP.SIZE | Changes the size of the Microsoft Excel application window |

Macro Sheets Only

## APP.MINIMIZE()

Equivalent to the Control Minimize command for the application window. Minimizes the Microsoft Excel window. For detailed information on the Control Minimize command, see online Help.

### Note

This function is only for Microsoft Excel for Windows or OS/2. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return the #N/A error value.

### Related Functions

| | |
|---|---|
| APP.ACTIVATE | Activates an application |
| APP.MAXIMIZE | Maximizes the Microsoft Excel application window |
| APP.MOVE | Moves the Microsoft Excel application window |
| APP.RESTORE | Restores the Microsoft Excel application window |
| APP.SIZE | Changes the size of the Microsoft Excel application window |

Macro Sheets Only

## APP.MOVE(*x_num,y_num*)
## APP.MOVE?(*x_num,y_num*)

Equivalent to the Control Move command for the application window. Moves the Microsoft Excel window. In Microsoft Excel for Windows, APP.MOVE returns the #VALUE! error value if the application window is already maximized. For detailed information on the Control Move command, see online Help.

### Note

This function is only for Microsoft Excel for Windows or OS/2. You can use this function in

macros created with Microsoft Excel for the Macintosh, but it will return FALSE.

***X_num*** specifies the horizontal position of the Microsoft Excel window measured in points from the left edge of your screen to the left side of the Microsoft Excel window.

***Y_num*** specifies the vertical position of the Microsoft Excel window measured in points from the top edge of your screen to the top of the Microsoft Excel window.

APP.MOVE?, the dialog-box form of the function, doesn't display a dialog box. Instead, it is equivalent to pressing ALT, SPACEBAR, M or to dragging the title bar with the mouse. This lets you move the window with the keyboard or mouse. If you specify *x_num* and/or *y_num* in the dialog-box form of the function, the window is moved according to the specified coordinates and you are left in move mode.

For more information on moving a window, see Chapter 1, "Microsoft Excel Basics," in the *Microsoft Excel User's Guide*.

### Example

The following formula moves the Microsoft Excel window 25 points from the left edge of your screen and 18 points from the top edge of your screen:

```
APP.MOVE(25,18)
```

### Related Functions

| | |
|---|---|
| APP.ACTIVATE | Activates an application |
| APP.MAXIMIZE | Maximizes the Microsoft Excel application window |
| APP.MINIMIZE | Minimizes the Microsoft Excel application window |
| APP.RESTORE | Restores the Microsoft Excel application window |
| APP.SIZE | Changes the size of the Microsoft Excel application window |

*Macro Sheets Only*

## APP.RESTORE( )

Equivalent to the Control Restore command for the application window. Restores the Microsoft Excel window to its previous size and location. For detailed information on the Control Restore command, see online Help.

### Note

This function is only for Microsoft Excel for Windows or OS/2. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return FALSE.

### Related Functions

| | |
|---|---|
| APP.ACTIVATE | Activates an application |
| APP.MAXIMIZE | Maximizes the Microsoft Excel application window |
| APP.MINIMIZE | Minimizes the Microsoft Excel application window |
| APP.MOVE | Moves the Microsoft Excel application window |
| APP.SIZE | Changes the size of the Microsoft Excel application window |

*Macro Sheets Only*

## APP.SIZE(*x_num,y_num*)
## APP.SIZE?(*x_num,y_num*)

Equivalent to the Control Size command for the application window. Changes the size of the Microsoft Excel window. For detailed information on the Control Size command, see online Help.

### Note

This function is only for Microsoft Excel for Windows or OS/2. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return FALSE.



***X_num*** specifies the width of the Microsoft Excel window in points.

***Y_num*** specifies the height of the Microsoft Excel window in points.

APP.SIZE?, the dialog-box form of the function, doesn't display a dialog box. Instead, it is equivalent to pressing ALT, SPACEBAR, S or to dragging a window border with the mouse. This lets you size the window with the keyboard or mouse. If you specify *x_num* and/or *y_num* in the dialog-box form of the function, the window is sized according to the specified coordinates and you are left in size mode.

For more information on sizing a window, see Chapter 1, "Microsoft Excel Basics," in the *Microsoft Excel User's Guide*.

### Related Functions

| | |
|---|---|
| APP.ACTIVATE | Activates an application |
| APP.MAXIMIZE | Maximizes the Microsoft Excel application window |
| APP.MINIMIZE | Minimizes the Microsoft Excel application window |
| APP.MOVE | Moves the Microsoft Excel application window |
| APP.RESTORE | Restores the Microsoft Excel application window |

*Macro Sheets Only*

## APPLY.NAMES(name_array,*ignore,use_rowcol, omit_col,omit_row,order_num,append_last*)
## APPLY.NAMES?(*name_array,ignore,use_rowcol, omit_col,omit_row,order_num,append_last*)

Equivalent to choosing the Formula Apply Names command (full menus). Replaces definitions with their respective names. If no names are defined in the current selection or if only one cell is selected on the current sheet, APPLY.NAMES returns the #VALUE! error value. Use APPLY.NAMES to replace references and values in formulas with names. For detailed information on the Formula Apply Names command, see online Help.

***Name_array*** is the name or names to apply as text elements in an array.

- To give more than one name as the argument, you must use an array. For example:

  ```
  APPLY.NAMES({"DataRange","CriteriaRange"})
  ```

- If the names indicated by the argument *name_array* have already replaced all of the appropriate references or values, the #VALUE! error value is returned.

The next four arguments correspond to check boxes and options in the Formula Apply Names dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Ignore*** corresponds to the Ignore Relative/Absolute check box.

***Use_rowcol*** corresponds to the Use Row And Column Names check box. If *use_rowcol* is FALSE, the next two arguments are ignored.

***Omit_col*** corresponds to the Omit Column Name If Same Column check box.

***Omit_row*** corresponds to the Omit Row Name If Same Row check box.

***Order_num*** determines which range name is listed first when a cell reference is replaced by a row-oriented and a column-oriented range name, as shown in the following table:

| *Order_num* | Order of range names |
|---|---|
| 1 | Row Column |
| 2 | Column Row |

***Append_last*** determines whether the names most recently defined are also replaced.

- If *append_last* is TRUE, Microsoft Excel replaces the definitions of the names in *name_array* and also replaces the definitions of the last names defined.

- If *append_last* is FALSE or omitted, Microsoft Excel replaces the definitions of the names in *name_array* only.

### Example

The following formula applies the names SalesOne, SalesTwo, and SalesThree:

```
APPLY.NAMES({"SalesOne","SalesTwo",
"SalesThree"},TRUE,TRUE,FALSE,TRUE,1,FALSE)
```

### Related Functions

| | |
|---|---|
| CREATE.NAMES | Creates names automatically from text in a related row or column |
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| LIST.NAMES | Pastes names and their associated information |

*Macro Sheets Only*

## APPLY.STYLE(style_text)
## APPLY.STYLE?(style_text)

Equivalent to choosing the Format Style command, selecting a style, and choosing OK. Applies a previously defined style to the current selection. For detailed information on the Format Style command, see online Help.

***Style_text*** is the name, as quoted text, of a previously defined style. If *style_text* is not defined, APPLY.STYLE returns the #VALUE! error value and interrupts the macro. If *style_text* is omitted, the Normal style is applied to the selection.

### Examples

The following macro formula applies a style named Titles to the current selection:

```
APPLY.STYLE("Titles")
```

To apply a style that exists for a particular cell to another range, use the GET.CELL function, with 40 as the *type_num* argument, to return the name of the style, as shown in the following macro:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | GetStyle | |
| 4 | Style | =GET.CELL(40) | Gets style name |
| 5 | | =SELECT("NewTitles") | Selects a new range |
| 6 | | =APPLY.STYLE(Style) | Applies style to selection |
| 7 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| DEFINE.STYLE | Defines a cell style |
| DELETE.STYLE | Deletes a style |
| MERGE.STYLES | Imports styles from another document into the active document |

## AREAS(reference)

Returns the number of areas in a reference. An area is a range of contiguous cells or a single cell.

***Reference*** is a reference to a cell or range of cells and can refer to multiple areas. If you want to specify several references as a single argument, then you must include extra sets of parentheses so that Microsoft Excel will not interpret the comma as a field separator. See the second example below.

### Tip

This function is useful in a macro for testing whether a reference is a multiple selection. See the fourth example below.

### Examples

```
AREAS(B2:D4) equals 1

AREAS((B2:D4,E5,F6:I9)) equals 3
```

If the name Prices refers to the areas B1:D4, B2, and E1:E10, then:

```
AREAS(Prices) equals 3
```

The following macro formula branches to an error routine if the current selection is not contiguous:

```
IF(AREAS(SELECTION())>1,GOTO(Error))
```



### Related Functions

| | |
|---|---|
| ADDRESS | Returns a reference to a single cell in a worksheet |
| CELL | Returns information about the formatting, location, or contents of a cell |
| COLUMN | Returns the column number of a reference |
| COLUMNS | Returns the number of columns in a reference |
| INDEX | Uses an index to choose a value from a reference or array |
| ROW | Returns the row number of a reference |
| ROWS | Returns the number of rows in a reference |

*Macro Sheets Only*

## ARGUMENT(name_text,data_type_num)
## ARGUMENT(name_text,data_type_num,reference)

Describes the arguments used in a custom function, which is a type of macro, or in a subroutine. A custom function or subroutine must contain one ARGUMENT function for each argument in the macro itself. There are two forms of the ARGUMENT function. In the first form, only *name_text* is required; in the second form, only *reference* is required. Use the first form if you want to store the argument as a name. Use the second form if you want to store the argument in a specific cell or cells.

For more information on custom functions and subroutines, see Chapter 17, "Creating and Using Custom Worksheet Functions," and Chapter 19, "Designing and Writing a Command Macro," in the *Microsoft Excel User's Guide*.

***Name_text*** is the name of the argument or of the cells containing the argument. *Name_text* is required if you omit *reference*.

***Data_type_num*** is a number that determines what type of values Microsoft Excel accepts for the argument. The following table lists the possible data types:

| *Data_type_num* | Type of value |
|---|---|
| 1 | Number |
| 2 | Text |
| 4 | Logical |
| 8 | Reference |
| 16 | Error |
| 64 | Array |

- *Data_type_num* can be a sum of the different numbers above to allow for more than one possible type of data. For example, if *data_type_num* is 7, which is the sum of 1, 2, and 4, then the value can be a number, text, or logical value.
- *Data_type_num* is an optional argument. If you omit *data_type_num*, it is assumed to be 7.
- If the value that is passed to the function macro is not of the type specified by *data_type_num*, Microsoft Excel first attempts to convert it to the specified type. If the value cannot be converted, the macro returns the #VALUE! error value.

***Reference*** is the cell or cells in which you want to store the argument's value.

- If you specify *reference*, the value that is passed to ARGUMENT is entered as a constant in the specified cell, and *name_text* becomes an optional argument because you can refer to the cell with either *reference* or *name_text*.
- If you omit *reference*, *name_text* is defined on the macro sheet and refers to the value that is passed to ARGUMENT. Once *name_text* is defined, you can use it in formulas. See the first example below.

### Notes

- Custom functions and subroutines can accept from 1 to 13 arguments.
- If a macro contains an ARGUMENT function and you omit the corresponding argument in the function that starts the macro, the macro uses the #N/A error value as the value of the argument.

### Examples

The following custom function calculates profit; each argument must be a number:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | PROFIT | Calculates profit |
| 4 | | CalculateProfit | |
| 5 | | =RESULT(1) | Returns a number |
| 6 | | =ARGUMENT("UnitsSold",1) | Defines the three |
| 7 | | =ARGUMENT("UnitCost",1) | arguments |
| 8 | | =ARGUMENT("UnitPrice",1) | |
| 9 | TotalProfit | =UnitsSold*(UnitPrice–UnitCost) | Computes the profit |
| 10 | | =RETURN(TotalProfit) | Returns the profit |

The following lines in a macro ask if you want to open a document. If you do, cell B25 calls a subroutine to make sure the document is available. The entire subroutine is shown, although only the last few lines of the calling macro are shown.

| | A | B | C |
|---|---|---|---|
| 22 | OpenYes | =ALERT("Open another document?",1) | |
| 23 | | =IF(OpenYes) | If OK, the macro continues |
| 24 | DocToOpen | =INPUT("Enter the document to open",2) | |
| 25 | | =CheckDocument(DocToOpen) | Calls subroutine |
| 26 | | =END.IF() | |
| 27 | | =RETURN() | |
| 28 | | | |
| 29 | | | |
| 30 | | CheckDocument | Checks for errors |
| 31 | | =ARGUMENT("NameOfDoc") | |
| 32 | | =ERROR(TRUE,ErrorOccurs) | Adds custom ALERT |
| 33 | | =OPEN(NameOfDoc) | Tries to open document |
| 34 | | =GOTO(End) | If successful, macro ends |
| 35 | ErrorOccurs | =ALERT(""&NameOfDoc&"" did not open, | Otherwise, error message |
| 36 | End | =RETURN() | |

The complete formula in cell B35 is:

```
=ALERT(""&NameOfDoc&"" did not open; it may not be in the current directory or folder",2)
```

### Related Function

RESULT — Specifies the type of value a custom function returns

*Macro Sheets Only*

## ARRANGE.ALL( )

Equivalent to the Window Arrange All command. Rearranges and resizes open windows so that you can see all of them at the same time. This is useful when you need to see how changes to one document affect other documents. For detailed information on the Window Arrange All command, see online Help.

### Related Function

ACTIVATE — Activates a window or a pane of a window

## ASIN(*number*)

Returns the arcsine of a number. The arcsine is the angle whose sine is *number*. The returned angle is given in radians in the range $-\pi/2$ to $\pi/2$.

***Number*** is the sine of the angle you want and must be from –1 to 1.

To express the arcsine in degrees, multiply the result by 180/PI( ).

### Examples

ASIN(–0.5) equals –0.524 ($-\pi/6$ radians)

ASIN(–0.5)*180/PI() equals –30 (degrees)

### Related Functions

ASINH — Returns the inverse hyperbolic sine of a number

PI — Returns the value π

SIN — Returns the sine of a number



## ASINH(*number*)

Returns the inverse hyperbolic sine of *number*. The inverse hyperbolic sine is the value whose hyperbolic sine is *number*, so ASINH(SINH(*number*)) equals *number*.

### Examples

ASINH(–2.5) equals –1.647231146

ASINH(10) equals 2.99822295

### Related Functions

ACOSH — Returns the inverse hyperbolic cosine of a number

ATANH — Returns the inverse hyperbolic tangent of a number

SINH — Returns the hyperbolic sine of a number

*Macro Sheets Only*

## ASSIGN.TO.OBJECT(*macro_ref*)
## ASSIGN.TO.OBJECT?(*macro_ref*)

Equivalent to the Macro Assign To Object command. Assigns a macro to be run when an object is clicked with the mouse. If an object is not selected, ASSIGN.TO.OBJECT returns the #VALUE! error value and interrupts the macro. For detailed information on the Macro Assign To Object command, see online Help.

***Macro_ref*** is a reference to a macro to be run when the object is clicked. *Macro_ref* can be a quoted reference to a macro or the name of a macro. If you omit *macro_ref*, no macro is assigned and any macro that had been previously assigned to the selected object will be unassigned.

### Note

To change the macro assigned to an object, select the object and use ASSIGN.TO.OBJECT again. The previous macro is replaced with the new macro.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula assigns to the selected object the macro MakeWorkGroup on the macro sheet WORKGRP.XLM:

```
ASSIGN.TO.OBJECT("WORKGRP.XLM!MakeWorkGroup")
```

In Microsoft Excel for the Macintosh, the following macro formula assigns to the selected object the macro MakeWorkGroup on the macro sheet Workgroups:

```
ASSIGN.TO.OBJECT("Workgroups!MakeWorkGroup")
```

### Related Functions

CREATE.OBJECT — Creates the specified object

RUN — Runs a macro

## ATAN(*number*)

Returns the arctangent of a number. The arctangent is the angle whose tangent is *number*. The returned angle is given in radians in the range $-\pi/2$ to $\pi/2$.

***Number*** is the tangent of the angle you want.

To express the arctangent in degrees, multiply the result by 180/PI( ).

### Examples

ATAN(1) equals 0.785 ($\pi/4$ radians)

ATAN(1)*180/PI() equals 45 (degrees)

### Related Functions

ATAN2 — Returns the arctangent from x- and y-coordinates

ATANH — Returns the inverse hyperbolic tangent of a number

PI — Returns the value π

TAN — Returns the tangent of a number

## ATAN2(x_num,y_num)

Returns the arctangent of the specified x- and y-coordinates. The arctangent is the angle from the x-axis to a line containing the origin (0,0) and a point with coordinates (*x_num*,*y_num*). The angle is given in radians between $-\pi$ and $\pi$, excluding $-\pi$.

**X_num** is the x-coordinate of the point.

**Y_num** is the y-coordinate of the point.

If both *x_num* and *y_num* are 0, ATAN2 returns the #DIV/0! error value.

To express the arctangent in degrees, multiply the result by 180/PI().

### Notes

- A positive result represents a counterclockwise angle from the x-axis; a negative result represents a clockwise angle.
- ATAN2(*a*,*b*) equals ATAN(*b*/*a*), except that *a* can equal 0 in ATAN2.

### Examples

ATAN2(1,1) equals 0.785 ($\pi$/4 radians)

ATAN2(-1,-1) equals -2.356 (-3$\pi$/4 radians)

ATAN2(-1,1) equals 1.57 (approximately $\pi$/2)

### Related Functions

| | |
|---|---|
| ATAN | Returns the arctangent of a number |
| ATANH | Returns the inverse hyperbolic tangent of a number |
| PI | Returns the value $\pi$ |
| TAN | Returns the tangent of a number |

## ATANH(number)

Returns the inverse hyperbolic tangent of *number*. *Number* must be between –1 and 1 (excluding –1 and 1). The inverse hyperbolic tangent is the value whose hyperbolic tangent is *number*, so ATANH(TANH(*number*)) equals *number*.

### Examples

ATANH(0.76159416) equals 1, approximately

ATANH(–0.1) equals –0.100335348

### Related Functions

| | |
|---|---|
| ACOSH | Returns the inverse hyperbolic cosine of a number |
| ASINH | Returns the inverse hyperbolic sine of a number |
| TANH | Returns the hyperbolic tangent of a number |

Macro Sheets Only

## ATTACH.TEXT(attach_to_num,series_num,point_num)

## ATTACH.TEXT?(attach_to_num,series_num,point_num)

Equivalent to the Chart Attach Text command. Attaches text to certain parts of the selected chart. Use ATTACH.TEXT with the FORMULA function to attach text as a title or as a label for an axis, plot text on data point. For detailed information on the Chart Attach Text command, see online Help.

**Attach_to_num** specifies which item on a chart to attach text to. *Attach_to_num* is different for 2-D and 3-D charts. *Attach_to_num* values for 2-D charts are shown in the following table:

| Attach_to_num | Attaches text to |
|---|---|
| 1 | Chart title |
| 2 | Value (y) axis |
| 3 | Category (x) axis |
| 4 | Series or data point |
| 5 | Overlay value (y) axis |
| 6 | Overlay category (x) axis |



*Attach_to_num* values for 3-D charts are shown in the following table:

| Attach_to_num | Attaches text to |
|---|---|
| 1 | Chart title |
| 2 | Value (z) axis |
| 3 | Series (y) axis |
| 4 | Category (x) axis |
| 5 | Series or data point |

**Series_num** specifies the series number if *attach_to_num* specifies a series or data point. If *attach_to_num* specifies a series or data point and *series_num* is omitted, the macro is interrupted.

**Point_num** specifies the number of the data point, but only if you specify a series number. *Point_num* is optional.

### Note

When you record adding an axis title or a chart title with the Chart Attach Text command, Microsoft Excel records both an ATTACH.TEXT function to attach the text and a FORMAT.FONT function to make the text bold.

### Example

The following macro to attach text to the selected chart

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **AddLabels** | Adds labels to a chart |
| 4 | X_Label | =INPUT("Enter x-axis label",2) | |
| 5 | | =ATTACH.TEXT(3) | |
| 6 | | =FORMULA(X_Label) | Labels the x-axis |
| 7 | Y_Label | =INPUT("Enter y-axis label",2) | |
| 8 | | =ATTACH.TEXT(2) | |
| 9 | | =FORMULA(Y_Label) | Labels the y-axis |
| 10 | TitleText | =INPUT("Enter a title",2) | |
| 11 | | =ATTACH.TEXT(1) | |
| 12 | | =FORMULA(TitleText) | Gives the chart a title |
| 13 | | =RETURN() | |

Macro Sheets Only

## ATTRIBUTES(comment_text,auto_share)

## ATTRIBUTES?(comment_text,auto_share)

Equivalent to the File Object Attributes command when you're working with Microsoft Excel objects in NewWave. Sets options in the File Object Attributes dialog box for the selected object.

### Note

This function is only available if you are using Microsoft Excel with NewWave.

**Comment_text** is optional text up to 255 characters long that you can use to describe the object.

**Auto_share** is a logical value corresponding to the Autoshared option in the File Object Attributes dialog box.

- If *auto_share* is TRUE, Microsoft Excel turns on the check box, and the object is shared automatically when you copy another object that displays it.
- If *auto_share* is FALSE, Microsoft Excel turns off the check box, and the object is not shared automatically when you copy an object that displays it.
- If *auto_share* is omitted, the autoshare status is unchanged.

*Comment_text* and *auto_share* are the only two options that you can set in the File Object Attributes dialog box. The others are determined automatically.

### Note

Copies of objects that display an autoshared object contain a shared view of the autoshared object, not an actual copy of it.

### Example

The following macro formula turns off the Autoshared option and adds the specified comment for the selected object.

ATTRIBUTES("Personal Budget for 1990",FALSE)

### Related Functions

| | |
|---|---|
| SAVE.NEW.OBJECT | Saves a copy of the Microsoft Excel object as a new object in NewWave |
| SHARE | Places a shared view of the current selection on the Clipboard in NewWave |
| SHARE.NAME | Places a shared view of a named range on the Clipboard in NewWave |

## AVERAGE(*number1,number2,...*)

Returns the average (mean) of the arguments.

***Number1, number2,...*** are 1 to 14 numeric arguments for which you want the average.

- The arguments should be numbers, or names, arrays, or references that contain numbers.
- If an array or reference argument contains text, logical values, or empty cells, those values are ignored; however, cells with the value zero are included.

### Note

When averaging cells, keep in mind the difference between empty cells and those containing the value zero, especially if you have turned off the Zero Values check box in the Options Display dialog box. Empty cells are not counted, but zero values are.

### Examples

If A1:A5 is named Scores and contains the numbers 10, 7, 9, 27, and 2, then:

AVERAGE(A1:A5) equals 11

AVERAGE(Scores) equals 11

AVERAGE(A1:A5,5) equals 10

AVERAGE(A1:A5) equals SUM(A1:A5)/COUNT(A1:A5) equals 11

If C1:C3 is named OtherScores and contains the numbers 4, 18, and 7, then:

AVERAGE(Scores,OtherScores) equals 10.5

To see AVERAGE working with other functions, see the example for REFTEXT.

### Related Functions

| | |
|---|---|
| COUNT | Counts how many numbers are in the list of arguments |
| COUNTA | Counts how many values are in the list of arguments |
| DAVERAGE | Returns the average of selected database entries |
| MEDIAN | Returns the median of numbers |
| SUM | Adds its arguments |

Macro Sheets Only

## AXES(*x_main,y_main,x_over,y_over*)
## AXES?(*x_main,y_main,x_over,y_over*)

**Form 1: 2-D charts**

## AXES(*x_main,y_main,z_main*)
## AXES?(*x_main,y_main,z_main*)

**Form 2: 3-D charts**

Equivalent to the Chart Axes command. Controls whether the axes on a chart are visible. There are two forms of this function. The first form is for 2-D charts; the second form is for 3-D charts. For detailed information on the Chart Axes command, see online Help.

Arguments are logical values corresponding to the check boxes in the Chart Axes dialog box.

- If an argument is TRUE, Microsoft Excel turns on the check box and displays the corresponding axis.
- If an argument is FALSE, Microsoft Excel turns off the check box and hides the corresponding axis.



***X_main*** corresponds to the x-axis on the main chart.

***Y_main*** corresponds to the y-axis on the main chart.

***Z_main*** corresponds to the z-axis on the 3-D main chart.

***X_over*** corresponds to the x-axis on the overlay chart.

***Y_over*** corresponds to the y-axis on the overlay chart.

If a 2-D chart has no overlay, only the first two arguments are used.

### Examples

The following macro formula displays the x-axis and y-axis on the selected 2-D chart:

AXES(TRUE,TRUE,FALSE,FALSE)

The following macro formula hides the x-axis on the selected 3-D chart:

AXES(FALSE,TRUE,TRUE)

### Related Function

| | |
|---|---|
| GRIDLINES | Controls whether major and minor gridlines are visible |

Macro Sheets Only

## BEEP(*tone_num*)

Sounds a tone. Use BEEP to signal a message, a dialog box, or the end of a macro, or whenever you need to get the user's attention.

***Tone_num*** is a number from 1 to 4 specifying a certain tone on some computers.

- On most computers, all numbers produce the same sound—the sound that you hear when an error occurs or when you click outside some dialog boxes.
- If *tone_num* is omitted, it is assumed to be 1.

### Notes

- On a Macintosh, you can control the volume of the tone by using the Control Panel desk accessory.
- In Windows 3.0 or OS/2 Presentation Manager, you can turn off the tone by using the Control Panel.

### Related Functions

| | |
|---|---|
| ALERT | Displays a dialog box with a message |
| MESSAGE | Displays a message in the status bar |

Macro Sheets Only

## BORDER(*outline,left,right,top,bottom,shade, outline_color,left_color,right_color,top_color, bottom_color*)
## BORDER?(*outline,left,right,top,bottom, shade,outline_color,left_color,right_color,top_color, bottom_color*)

Equivalent to the Format Border command. Adds a border to the selected cell, range of cells, or object. For detailed information on the Format Border command, see online Help.

***Outline***, ***left***, ***right***, ***top***, and ***bottom*** are numbers from 0 to 7 corresponding to the line styles in the Format Border dialog box, as shown below.

| Argument | Line type |
|---|---|
| 0 | No border |
| 1 | Thin line |
| 2 | Medium line |
| 3 | Dashed line |
| 4 | Dotted line |
| 5 | Thick line |
| 6 | Double line |
| 7 | Hairline |

*Note*

For compatibility with earlier versions of Microsoft Excel, TRUE and FALSE values for the above arguments create a single line border or no border, respectively.

***Outline_color, left_color, right_color, top_color,*** and ***bottom_color*** are numbers from 1 to 16 corresponding to the Color box in the Format Border dialog box. Zero corresponds to automatic color.

### Example

The following macro formula puts a dashed line around the top, bottom, left, and right of each cell in the selection, using color 5 for the left and right and color 10 for the top and bottom.

```
BORDER(0,3,3,3,3,,,5,5,10,10)
```

### Related Function

| | |
|---|---|
| PATTERNS | Changes the appearance of the selected object |

*Macro Sheets Only*

## BREAK()

Interrupts a FOR–NEXT, a FOR.CELL–NEXT, or a WHILE–NEXT loop. If BREAK is encountered within a loop, that loop is terminated and the macro proceeds to the statement following the NEXT statement at the end of the current loop. For information on looping, see Chapter 19, "Designing and Writing a Command Macro," in the *Microsoft Excel User's Guide*.

### Example

Use BREAK to test for conditions not anticipated by the FOR or WHILE statement. For example, the following macro uses BREAK nested in an IF statement to exit a WHILE–NEXT loop when a certain value is encountered.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **ExitLoop** | |
| 4 | | =SET.VALUE(Counter,1) | Initializes Counter to 1 |
| 5 | | =WHILE(Counter<10) | Begins WHILE–NEXT loop |
| 6 | Counter | =Counter+INT(RAND()*5) | Increments Counter |
| 7 | | =ProcessCounter(Counter) | Runs subroutine |
| 9 | | =IF(Counter=8,BREAK()) | If Counter = 8, break |
| 9 | | =NEXT() | Ends WHILE–NEXT loop |
| 10 | | =RETURN() | Ends the macro |

### Related Functions

| | |
|---|---|
| FOR | Begins a FOR–NEXT loop |
| FOR.CELL | Begins a FOR.CELL–NEXT loop |
| NEXT | Ends a FOR–NEXT, FOR.CELL–NEXT, or WHILE–NEXT loop |
| WHILE | Begins a WHILE–NEXT loop |

*Macro Sheets Only*

## BRING.TO.FRONT()

Equivalent to the Format Bring To Front command. Puts the selected object or objects on top of all other objects. For example, if some worksheet objects are covering part of an embedded chart, you can select the chart and use BRING.TO.FRONT to display the chart on top of the worksheet objects. For detailed information on the Format Bring To Front command, see online Help.

If the selection is not an object or a group of objects, BRING.TO.FRONT returns the #VALUE! error value.

### Related Function

| | |
|---|---|
| SEND.TO.BACK | Sends selected objects behind other worksheet objects |



*Macro Sheets Only*

## CALCULATE.DOCUMENT()

Equivalent to the Options Calculate Document command. Calculates the active document. Use CALCULATE.DOCUMENT to calculate only the active worksheet. For detailed information on the Options Calculate Document command, see online Help.

### Note

If the active document is a chart, CALCULATE.DOCUMENT returns the #VALUE! error value.

### Related Functions

| | |
|---|---|
| CALCULATE.NOW | Calculates all open documents immediately |
| CALCULATION | Controls calculation settings |

*Macro Sheets Only*

## CALCULATE.NOW()

Equivalent to the Options Calculate Now and Chart Calculate Now commands. Calculates all open documents. Use CALCULATE.NOW to calculate all open documents when Options Calculation is set to Manual. For detailed information on the Options Calculate Now command, see online Help.

### Related Functions

| | |
|---|---|
| CALCULATE.DOCUMENT | Calculates the active document only |
| CALCULATION | Controls calculation settings |

*Macro Sheets Only*

## CALCULATION(*type_num,iter,max_num, max_change,update,precision,date_1904,calc_save, save_values*)

## CALCULATION?(*type_num,iter,max_num, max_change,update,precision,date_1904,calc_save, save_values*)

Equivalent to the Options Calculation command. Use CALCULATION to control when and how formulas in open documents are calculated. For detailed information on the Options Calculation command, see online Help.

Arguments correspond to check boxes and options in the Options Calculation dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Type_num*** is a number from 1 to 3 indicating the type of calculation, as follows:

| *Type_num* | Type of calculation |
|---|---|
| 1 | Automatic |
| 2 | Automatic except tables |
| 3 | Manual |

***Iter*** corresponds to the Iteration check box. The default is FALSE.

***Max_num*** is the maximum number of iterations. The default is 100.

***Max_change*** is the maximum change of each iteration. The default is 0.001.

***Update*** corresponds to the Update Remote References check box. The default is TRUE.

***Precision*** corresponds to the Precision As Displayed check box. The default is FALSE.

***Date_1904*** corresponds to the 1904 Date System check box. The default is FALSE in Microsoft Excel for Windows or OS/2 and TRUE in Microsoft Excel for the Macintosh.

***Calc_save*** is equivalent to the Recalculate Before Save check box. If *calc_save* is FALSE, the document is not recalculated before saving when in manual calculation mode. The default is TRUE.

***Save_values*** corresponds to the Save External Link Values check box. The default is TRUE.

*Note*

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

### Related Functions

| | |
|---|---|
| CALCULATE.DOCUMENT | Calculates the active document only |
| CALCULATE.NOW | Calculates all open documents immediately |

## CALL(*register_ref,argument1,...*)

**Form 1: Used with REGISTER**

## CALL(*module_text,procedure_text,type_text, argument1,...*)

**Form 2a: Used alone (in Microsoft Excel for Windows or OS/2)**

## CALL(*file_text,resource_text,type_text, argument1,...*)

**Form 2b: Used alone (in Microsoft Excel for the Macintosh)**

*Important*

This function is provided for advanced users only. If you use the CALL function incorrectly you could accidentally cause errors in your system's operation.

Calls a procedure in a dynamic link library or code resource. There are two forms of this function. The first form can only be used with a code resource registered by the REGISTER function. The second form uses the arguments to the REGISTER function and can be used on worksheets to simultaneously register and call a code resource. For more information and for descriptions of each argument to this form, see REGISTER.

***Register_ref*** is the value returned by a previously executed REGISTER function. You give REGISTER the name of the module and procedure you want to access, as well as a text value that describes the number and data types of arguments you want to give and the data type of the return value. REGISTER then returns the value of *register_ref* to use in the CALL function.

***Argument1,...*** are the arguments to be given to the procedure.

See the REGISTER function and Appendix A, "Using the CALL and REGISTER Functions," for descriptions of the other arguments and for information on the data types and how they are given and returned.

### Examples

The example for the REGISTER function shows how to use REGISTER to get the *register_ref* value for calling the GetSysColor procedure. If the result of that REGISTER function was in cell A5, you could use the following CALL formula to call the GetSysColor procedure, giving the argument 2:

```
CALL(A5,2)
```

The following formula on a worksheet registers the GetSysColor procedure:

```
CALL("User","GetSysColor","JH",2)
```

### Related Functions

| | |
|---|---|
| REGISTER | Returns a number to be used by the CALL function |
| UNREGISTER | Removes a resource that was previously registered by the REGISTER function |

*Macro Sheets Only*

## CALLER()

Can be used in combination with other value-returning functions to return information about the cell or range of cells that called the custom function that is currently running. CALLER can only be used inside a custom function (also known as a function macro) or an Auto_Open or Auto_Close macro.

- If the function is entered in a single cell, CALLER returns the reference of that cell.
- If the function was part of an array formula entered into a range of cells, CALLER returns the reference of the range.
- If CALLER appears in a macro called by an Auto_Open or Auto_Close macro, it returns the name of the sheet calling the Auto_Open or Auto_Close macro.
- If CALLER appears in a macro called by an assigned-to-object macro, it returns the object identifier.
- If the currently running macro is a command macro started by the user, CALLER returns the #REF! error value.

*Note*

CALLER is useful in a custom function whose behavior depends on the location or size of the reference that started the macro.

### Examples

If the custom function Macros!ValueOne is entered in cell B3 on a worksheet named SALES, the nested CALLER function returns the following values:

| Nested function | Returns |
|---|---|
| COLUMN(CALLER()) | 2 |
| COLUMNS(CALLER()) | 1 |
| GET.CELL(1,CALLER()) | SALES!$B$3 |
| REFTEXT(CALLER()) | SALES!R3C2 |
| ROW(CALLER()) | 3 |
| ROWS(CALLER()) | 1 |

If the same custom function was entered into an array in cells B2:C3, the following values would be returned:

| Nested function | Returns |
|---|---|
| COLUMN(CALLER()) | 2 |
| COLUMNS(CALLER()) | 2 |
| GET.CELL(1,CALLER()) | SALES!$B$2 |
| REFTEXT(CALLER()) | SALES!R2C2:R3C3 |
| ROW(CALLER()) | 2 |
| ROWS(CALLER()) | 2 |

### Related Functions

| | |
|---|---|
| COLUMN | Returns the number of the active column |
| COLUMNS | Returns the number of columns in the selection |
| GET.CELL | Returns information about a cell |
| REFTEXT | Converts a reference to text |
| ROW | Returns the number of the active row |
| ROWS | Returns the number of rows in the selection |



*Macro Sheets Only*

## CANCEL.COPY()

Equivalent to pressing ESC in Microsoft Excel for Windows or OS/2 or ESC or COMMAND+PERIOD in Microsoft Excel for the Macintosh to cancel the marquee after you copy or cut a selection.

*Macro Sheets Only*

## CANCEL.KEY(*enable,macro_ref*)

Disables macro interruption, or specifies a macro to run when a macro is interrupted. Use CANCEL.KEY to control what happens when a macro is interrupted.

***Enable*** specifies whether the macro can be interrupted by pressing ESC in Microsoft Excel for Windows or OS/2 or ESC or COMMAND+PERIOD in Microsoft Excel for the Macintosh.

| If *enable* is | Then |
|---|---|
| FALSE or omitted | Pressing ESC or COMMAND+PERIOD does not interrupt a macro |
| TRUE and *macro_ref* is omitted | Pressing ESC or COMMAND+PERIOD interrupts a macro |
| TRUE and *macro_ref* is specified | *Macro_ref* runs when ESC or COMMAND+PERIOD is pressed |

***Macro_ref*** is a reference to a macro, as a cell reference or a name, that runs when *enable* is TRUE and ESC or COMMAND+PERIOD is pressed.

### Note

CANCEL.KEY affects only the macro that is currently running. Once the macro is stopped by a RETURN or HALT function, ESC or COMMAND+PERIOD is reactivated.

### Examples

The following formula prevents the macro from being interrupted by pressing ESC or COMMAND+PERIOD:

```
CANCEL.KEY(FALSE)
```

The following formula reactivates ESC or COMMAND+PERIOD to cancel macro execution:

```
CANCEL.KEY(TRUE)
```

The following line in a macro named CheckCancel runs CheckCancel when ESC or COMMAND+PERIOD is pressed:

```
CANCEL.KEY(TRUE,CheckCancel())
```

### Related Functions

| | |
|---|---|
| ERROR | Specifies an action to take if an error occurs while the macro is running |
| ON.KEY | Runs a macro when a specified key is pressed |
| ON.TIME | Runs a macro at a specified time |

## CELL(*info_type*,*reference*)

Returns information about the formatting, location, or contents of the upper-left cell in a reference.

### Note

The CELL function is provided for compatibility with other spreadsheet programs. If you need to use cell information in a macro, GET.CELL provides a broader set of attributes.

***Info_type*** is a text value that specifies what type of cell information you want. The following list shows the possible values of *info_type* and the corresponding results.

| *Info_type* | Returns |
|---|---|
| "width" | Column width of the cell rounded off to an integer. Each unit of column width is equal to the width of one character in the currently selected font size. |
| "row" | Row number of the cell in *reference*. |
| "col" | Column number of the cell in *reference*. |
| "protect" | 0 if the cell is not locked, and 1 if the cell is locked. |
| "address" | Reference of the first cell in *reference*, as text. |
| "contents" | Contents of the upper-left cell in *reference*. |
| "format" | Text value corresponding to the number format of the cell. The text values for the various formats are shown in a table below. Returns "-" at the end of the text value if the cell is formatted for color for negative values. Returns "()" at the end of the text value if the cell is formatted for parentheses for positive or all values. |



| *Info_type* | Returns |
|---|---|
| "prefix" | Text value corresponding to the "label prefix" of the cell. Returns single quotation mark (') if the cell contains left-aligned text, double quotation mark (") if the cell contains right-aligned text, caret (^) if the cell contains centered text, backslash (\) if the cell contains fill-aligned text, and empty text ("") if the cell contains anything else. |
| "color" | 1 if the cell is formatted for color for negative values; otherwise returns 0. |
| "filename" | Filename (including full path) of the file that contains *reference*, as text. Returns empty text ("") if the worksheet that contains *reference* has not yet been saved. |
| "parentheses" | 1 if the cell is formatted for parentheses for positive or all values; otherwise returns 0. |
| "type" | Text value corresponding to the type of data in the cell. Returns "b" for blank if the cell is empty, "l" for label if the cell contains a text constant, and "v" for value if the cell contains anything else. |

***Reference*** is the cell that you want information about.

- If *reference* is omitted, it is assumed to be the active cell.
- If *reference* is a multiple selection, CELL returns the #VALUE! error value.

The following list describes the text values CELL returns when *info_type* is "format" and *reference* is a cell formatted with a built-in number format.

| If the Microsoft Excel format is | CELL returns |
|---|---|
| General | G |
| 0 | F0 |
| #,##0 | ,0 |
| 0.00 | F2 |
| #,##0.00 | ,2 |
| $#,##0_);($#,##0) | C0 |
| $#,##0_);[Red]($#,##0) | C0– |
| $#,##0.00_);($#,##0.00) | C2 |
| $#,##0.00_);[Red]($#,##0.00) | C2– |
| 0% | P0 |
| 0.00% | P2 |
| 0.00E+00 | S2 |
| # ?/? or # ??/?? | G |
| m/d/yy or m/d/yy h:mm | D4 |
| d-mmm-yy | D1 |
| d-mmm | D2 |
| mmm-yy | D3 |
| h:mm AM/PM | D7 |
| h:mm:ss AM/PM | D6 |
| h:mm | D9 |
| h:mm:ss | D8 |

If *info_type* is "format" and *reference* is formatted with a custom number format, Microsoft Excel returns a code representing the cell format. For more information on formats, see Chapter 5, "Formatting a Worksheet," in the *Microsoft Excel User's Guide*.

### Examples

CELL("row",A20) equals 20

If B12 has the format "d-mmm", then:

CELL("format",B12) equals "D2"

If the active cell contains TOTAL, then:

CELL("contents") equals "TOTAL"

### Related Function

| | |
|---|---|
| GET.CELL | Returns information about a cell |

Macro Sheets Only

## CELL.PROTECTION(*locked,hidden*)
## CELL.PROTECTION?(*locked,hidden*)

Equivalent to the Format Cell Protection command (full menus). Allows you to control cell protection and display. For detailed information on the Format Cell Protection command, see online Help.

Arguments are logical values corresponding to check boxes in the Format Cell Protection dialog box. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box. If an argument is omitted and the setting has been previously changed from the defaults, the setting is not changed.

***Locked*** corresponds to the Locked check box. The default is TRUE.

***Hidden*** corresponds to the Hidden check box. The default is FALSE.

### Note

Options selected in the Format Cell Protection dialog box or with the CELL.PROTECTION function are activated only when the Options Protect Document command is chosen or the PROTECT.DOCUMENT function is used to turn on protection.

### Examples

The following macro formula locks the selected cells when the Options Protect Document command is chosen:

```
CELL.PROTECTION(TRUE)
```

The following macro formula unlocks and hides the selected cells when the Options Protect Document command is chosen:

```
CELL.PROTECTION(FALSE,TRUE)
```

### Related Functions

| | |
|---|---|
| PROTECT.DOCUMENT | Turns on and off cell and window protection |
| SAVE.AS | Allows a password to be specified for a file to be opened |

Macro Sheets Only

## CHANGE.LINK(*old_text,new_text,type_of_link*)
## CHANGE.LINK?(*old_text,new_text,type_of_link*)

Equivalent to choosing the Change button in the File Links dialog box (full menus). Changes a link from one supporting document to another. For more information on the File Links command, see online Help.

***Old_text*** is the pathname of the link from the active dependent document you want to change.

***New_text*** is the pathname of the link you want to change to.

***Type_of_link*** is the number 1 or 2 specifying what type of link you want to get information about.

| *Type_of_link* | Link document type |
|---|---|
| 1 | Microsoft Excel link |
| 2 | DDE link |

### Note

The document whose links you want to change must be active when this function is calculated.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula changes the link on the active document from SALESET.XLS in the current directory to SALESWE.XLS in the specified directory:

```
CHANGE.LINK("SALESET.XLS","C:\EXCEL\SALESWE.XLS")
```

In Microsoft Excel for the Macintosh, the following macro formula changes the link on the active document from Sales East in the current folder to Sales West in the specified folder:

```
CHANGE.LINK("Sales East","Hard Drive:Excel:Sales West")
```

### Related Functions

| | |
|---|---|
| GET.LINK.INFO | Returns information about a link |
| OPEN.LINKS | Opens a specified supporting document |
| SET.UPDATE.STATUS | Sets the update status of a link to automatic or manual |
| UPDATE.LINK | Updates a link to another document |



## CHAR(*number*)

Returns the character corresponding to the code *number*. Use CHAR to translate code numbers you might get from files on other types of computers into characters.

***Number*** is a number between 1 and 255 specifying which character you want. The character is from the character set used by your computer.

| Operating environment | Character set |
|---|---|
| Macintosh | Macintosh character set |
| OS/2 | Code Page 850 |
| Windows | ANSI |

### Examples

CHAR(65) equals "A"

CHAR(33) equals "!"

### Related Function

| | |
|---|---|
| CODE | Returns the ANSI, Code Page 850, or Macintosh character set code for a character |

Macro Sheets Only

## CHECK.COMMAND(*bar_num,menu,command,check*)

Adds or removes a check mark from a command name on a menu. A check mark beside a command indicates that the command has been chosen.

***Bar_num*** is the menu bar containing the command. *Bar_num* can be the ID number of a built-in or custom menu bar.

***Menu*** is the menu containing the command. *Menu* can be either the name of a menu as text or the number of a menu. Menus are numbered starting with 1 from the left of the screen.

***Command*** is the command you want to check. *Command* can be the name of the command as text or the number of the command; the first command on a menu is in position 1.

***Check*** is a logical value corresponding to the check mark. If *check* is TRUE, Microsoft Excel adds a check mark to the command; if FALSE, Microsoft Excel removes the check mark.

### Note

The check mark doesn't affect execution of the command. Microsoft Excel automatically adds and deletes check marks to some commands, such as the name of the active document in the Window menu. If you have assigned a check mark to a built-in command that Microsoft Excel automatically changes in response to the user's actions, the check mark will be added or removed as appropriate and any check marks you have added or deleted with CHECK.COMMAND will be ignored.

### Example

The following macro formula adds a check mark to the Sales command on the Weekly menu on a custom menu bar created by the ADD.BAR function in a cell named Reports:

```
CHECK.COMMAND(Reports,"Weekly","Sales",TRUE)
```

### Related Functions

| | |
|---|---|
| ADD.COMMAND | Adds a command to a menu |
| DELETE.COMMAND | Deletes a command from a menu |
| ENABLE.COMMAND | Enables or disables a command |
| RENAME.COMMAND | Renames a command |

## CHOOSE(index_num,value1,value2,...)

Uses *index_num* to return a value from the list of value arguments. Use CHOOSE to select one of up to 13 values based on the index number. For example, if *value1* through *value7* are the days of the week, CHOOSE returns one of the days when a number between 1 and 7 is used as *index_num*.

***Index_num*** specifies which value argument is selected. *Index_num* must be a number between 1 and 13, or a formula or reference to a cell containing a number between 1 and 13.

- If *index_num* is 1, CHOOSE returns *value1*; if it is 2, CHOOSE returns *value2*; and so on.
- If *index_num* is less than 1 or greater than the number of the last value in the list, CHOOSE returns the #VALUE! error value.
- If *index_num* is a fraction, it is rounded to the nearest integer before being used.

***Value1, value2,...*** are 1 to 13 value arguments from which CHOOSE selects a value or an action to perform based on *index_num*. The arguments can be numbers, cell references, defined names, formulas, macro functions, or text. For information on using value arguments, see "Converting Data Types" earlier in this book.

### Notes

- If you are using CHOOSE in a macro, the *value* arguments can be GOTO functions or action-taking functions. For example, the following formulas are allowed in a macro, where Level is a name referring to a value or a cell containing a value between 1 and 3:

  ```
  CHOOSE(Level,GOTO(Begin),GOTO(Intermed),
  GOTO(Adv))
  ```

  ```
  CHOOSE(File,ACTIVATE.NEXT(),ACTIVATE.PREV())
  ```

- If *index_num* is an array, every *value* is evaluated when CHOOSE is executed. If some of those *value* arguments are action-taking functions, all the actions are taken. For example, the following formula opens both a new worksheet and a new chart:

  ```
  CHOOSE({1,2},NEW(1),NEW(2))
  ```

- The *value* arguments to CHOOSE can be range references as well as single values. For example, the formula:

  ```
  SUM(CHOOSE(2,A1:A10,B1:B10,C1:C10))
  ```

  evaluates to:

  ```
  SUM(B1:B10)
  ```

  which then returns a value based on the values in the range B1:B10.

  The CHOOSE function is evaluated first, returning the reference B1:B10. The SUM function is then evaluated using B1:B10, the result of the CHOOSE function, as its argument.

### Examples

```
CHOOSE(2,"1st","2nd","3rd","Finished") equals "2nd"
```

```
SUM(A1:CHOOSE(3,A10,A20,A30)) equals
SUM(A1:A30)
```

If A10 contains 4, then:

```
CHOOSE(A10,"Nails","Screws","Nuts","Bolts") equals
"Bolts"
```

If A10–3 equals 3, then:

```
CHOOSE(A10–3,"1st","2nd","3rd","Finished") equals "3rd"
```

If SalesOld is a name defined to refer to the value 10,000, then:

```
CHOOSE(2,SalesNew,SalesOld,SalesBudget) equals
10,000
```

### Related Function

| | |
|---|---|
| INDEX | Uses an index to choose a value from a reference or an array |

## CLEAN(text)

Removes all nonprintable characters from *text*. Use CLEAN on text imported from other applications which contains characters that may not print with your operating system. For example, you can use CLEAN to remove some low-level computer code that is frequently at the beginning and end of data files and cannot be printed.



***Text*** is any worksheet information from which you want to remove nonprintable characters.

### Example

Since CHAR(7) returns a nonprintable character:

```
CLEAN(CHAR(7)&"text"&CHAR(7)) equals "text"
```

### Related Functions

| | |
|---|---|
| CHAR | Returns the character specified by the code number |
| TRIM | Removes spaces from text |

*Macro Sheets Only*

## CLEAR(type_num)
## CLEAR?(type_num)

Equivalent to the Edit Clear command. Clears formulas, formats, notes, or all of these from the active worksheet or macro sheet. Clears formulas, formats, or both from the active chart. For detailed information on the Edit Clear command, see online Help.

***Type_num*** is a number from 1 to 4 specifying what to clear (only values 1, 2, and 3 are valid if the selected item is a chart). If you omit *type_num*, it is assumed to be 3 for a worksheet but 1 for a chart. On a worksheet or macro sheet or if the entire chart is selected:

| *Type_num* | Clears |
|---|---|
| 1 | All |
| 2 | Formats (if a chart, clears the chart format or clears pictures) |
| 3 | Formulas (if a chart, clears all data series) |
| 4 | Notes |

On a chart, if a single point or an entire data series is selected:

| *Type_num* | Clears |
|---|---|
| 1 | Selected series |
| 2 | Format in the selected point or series |

### Example

The following macro formula clears cell contents from the selected range of cells without affecting formats or notes:

```
CLEAR(3)
```

### Related Function

| | |
|---|---|
| DELETE | Removes cells from a worksheet |

*Macro Sheets Only*

## CLOSE(save_logical)

Closes the active window. In Microsoft Excel for Windows or OS/2, CLOSE is equivalent to the Close command on the document Control menu. In Microsoft Excel for the Macintosh, CLOSE is equivalent to clicking the close box.

***Save_logical*** is a logical value that specifies whether to save the file before closing the window.

| *Save_logical* | Result |
|---|---|
| TRUE | Saves the file |
| FALSE | Does not save the file |
| Omitted | If you've made changes to the file, displays a dialog box asking if you want to save the file |

### Note

For users of earlier versions of Microsoft Excel, if the macro sheet containing the function is the active sheet, CLOSE now closes the macro sheet.

### Example

The following macro closes all windows, except the main window, on the active document. If only one window is open on the document, the macro takes no action:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **CloseExtraWindows** | Closes all extra windows |
| 4 | Windows | =GET.DOCUMENT(13) | Number of open windows |
| 5 | | =IF(Windows>1) | |
| 6 | | =FOR("Counter",1,Windows-1) | Loops for all windows but one |
| 7 | | =CLOSE() | |
| 8 | | =NEXT() | |
| 9 | | =END.IF() | |
| 10 | | =RETURN() | |

***Related Functions***

| | |
|---|---|
| CLOSE.ALL | Closes all unprotected windows |
| FILE.CLOSE | Closes the active document |
| SAVE | Saves the active document |

*Macro Sheets Only*

## CLOSE.ALL()

Equivalent to the File Close All command. Closes all unprotected windows and all hidden windows. If unsaved changes have been made to the document in one or more windows, a message is displayed asking if you want to save the document. For detailed information on the File Close All command, see online Help.

***Related Functions***

| | |
|---|---|
| CLOSE | Closes the active window |
| FILE.CLOSE | Closes the active document |
| QUIT | Ends a Microsoft Excel session |
| SAVE | Saves the active document |

## CODE(*text*)

Returns a numeric code for the first character in a text string. The returned code corresponds to the character set used by your computer.

| Operating environment | Character set |
|---|---|
| Macintosh | Macintosh character set |
| OS/2 | Code Page 850 |
| Windows | ANSI |

***Text*** is the text for which you want the code of the first character.

***Examples***

CODE("A") equals 65

CODE("Alphabet") equals 65

The following macro returns the numeric code for each character in a string of text and places the codes in successive cells in the same column:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **CharacterCodes** | Returns codes for text |
| 4 | TextString | =INPUT("Enter your text:",2) | |
| 5 | | =FOR("Counter",1,LEN(TextString)) | Iterates for each character |
| 6 | CharCode | =CODE(MID(TextString,Counter,1)) | Code of current character |
| 7 | | =FORMULA(CharCode) | Places code in active cell |
| 8 | | =SELECT(OFFSET(ACTIVE.CELL(),1,0)) | Continues with next cell |
| 9 | | =NEXT() | |
| 10 | | =RETURN() | |

Also see the example for FREAD.

***Related Function***

| | |
|---|---|
| CHAR | Returns the character specified by the code number |

*Macro Sheets Only*

## COLOR.PALETTE(*file_text*)
## COLOR.PALETTE?(*file_text*)

Equivalent to choosing a file from the Copy Colors From box in the Options Color Palette dialog box. Copies a color palette from an open document to the active document. Use COLOR.PALETTE to share color palettes between documents. For detailed information on the Options Color Palette command, see online Help.

***File_text*** is the name of a document, as a quoted text string, that you want to copy a color palette from. The document specified by *file_text* must be open, or COLOR.PALETTE returns the #VALUE! error value and interrupts the macro. If *file_text* is omitted, then COLOR.PALETTE sets colors to the default values.

***Examples***

In Microsoft Excel for Windows or OS/2, the following macro formula copies the color palette from a document named MASTER.XLS to the active document:

COLOR.PALETTE("MASTER.XLS")

In Microsoft Excel for the Macintosh, the following macro formula copies the color palette from a document named Master to the active document:

COLOR.PALETTE("Master")

***Related Function***

| | |
|---|---|
| EDIT.COLOR | Defines a color on the color palette |

## COLUMN(*reference*)

Returns the column number of the given reference.

***Reference*** is the cell or range of cells for which you want the column number.

- If *reference* is omitted, it is assumed to be the reference of the cell in which the COLUMN function appears.
- If *reference* is a range of cells and if COLUMN is entered as a horizontal array, COLUMN returns the column numbers of *reference* as a horizontal array.
- *Reference* cannot refer to multiple areas.

***Examples***

COLUMN(A3) equals 1

When entered as an array in any three horizontally contiguous cells:

COLUMN(A3:C5) equals {1,2,3}

If COLUMN is entered in C5, then:

COLUMN() equals COLUMN(C5) equals 3

***Related Functions***

| | |
|---|---|
| COLUMNS | Returns the number of columns in a reference |
| ROW | Returns the row number of a reference |

*Macro Sheets Only*

## COLUMN.WIDTH(*width_num,reference,standard,type_num*)
## COLUMN.WIDTH?(*width_num,reference,standard,type_num*)

Equivalent to the Format Column Width command. Changes the width of the columns in the specified reference. For detailed information on the Format Column Width command, see online Help.

***Width_num*** specifies how wide you want the columns to be in units of one character of the font corresponding to the Normal cell style. *Width_num* is ignored if *standard* is TRUE or if *type_num* is provided.

***Reference*** specifies the columns for which you want to change the width.

- If *reference* is omitted, it is assumed to be the current selection.
- If *reference* is specified, it must be either an external reference to the active worksheet, such as !$A:$C or !Database, or a quoted R1C1-style reference in the form of text, such as "C1:C3", "C[–4]:C[–2]" or "Database".
- If *reference* is a relative R1C1-style reference in the form of text, it is assumed to be relative to the active cell.

***Standard*** is a logical value corresponding to the Standard Width check box in the Format Column Width dialog box.

- If *standard* is TRUE, Microsoft Excel sets the column width to the standard (default) width and ignores *width_num*.
- If *standard* is FALSE or omitted, Microsoft Excel sets the width according to *width_num* or *type_num*.

***Type_num*** is a number from 1 to 3 corresponding to the Hide, Unhide, or Best Fit button, respectively, in the Format Column Width dialog box.

| *Type_num* | Action taken |
|---|---|
| 1 | Hides the column selection by setting the column width to 0 |
| 2 | Unhides the column selection by setting the column width to the value set before the selection was hidden |
| 3 | Sets the column selection to a best-fit width, which varies from column to column depending on the length of the longest data string in each column |

#### Notes

- If any of the argument settings conflict, such as when *standard* is TRUE and *type_num* is 3, Microsoft Excel uses the *type_num* argument and ignores any arguments that conflict with *type_num*. See the second example below.
- If you are recording a macro while using a mouse and you change column widths by dragging the column border, Microsoft Excel records the references of the columns using R1C1 style references in the form of text.

#### Examples

If the range C1:E50 is selected, the following macro formula changes the width of columns C through E to 15:

```
COLUMN.WIDTH(15)
```

Each of the following macro formulas changes the width of columns C and D to the best fit:

```
COLUMN.WIDTH(,!$C:$D,,3)
COLUMN.WIDTH(,"C3:C4",TRUE,3)
```

The following macro formula hides the selected columns:

```
COLUMN.WIDTH(,,,1)
```

#### Related Function

| | |
|---|---|
| ROW.HEIGHT | Changes the heights of rows |

## COLUMNS(*array*)

Returns the number of columns in an array or reference.

***Array*** is an array or array formula, or a reference to a range of cells for which you want the number of columns.

#### Examples

```
COLUMNS(A1:C4) equals 3

COLUMNS({1,2,3;4,5,6}) equals 3
```

The following macro formula returns the number of columns in a contiguous selection:

```
COLUMNS(SELECTION())
```

If the selection is not contiguous, the following macro formula returns the number of columns in a particular area specified by AreaNum of the selection. An "area" is a contiguous region within a multiple selection, or a multiple selection consists of multiple areas.

```
COLUMNS(INDEX(SELECTION(),,,AreaNum))
```

#### Related Functions

| | |
|---|---|
| COLUMN | Returns the column number of a reference |
| ROWS | Returns the number of rows in a reference |



Macro Sheets Only

## COMBINATION(*type_num*)
## COMBINATION?(*type_num*)

Equivalent to the Gallery Combination command (full menus). Changes the format of the active chart to the combination format you select from the gallery. For detailed information on the Gallery Combination command, see online Help.

***Type_num*** is a number corresponding to the chart number in the Gallery Combination dialog box.

#### Example

The following macro formula formats both the main chart and overlay chart as line charts:

```
COMBINATION(3)
```

#### Related Function

| | |
|---|---|
| FORMAT.MAIN | Formats the selected main chart |

Macro Sheets Only

## CONSOLIDATE(*source_refs,function_num, top_row,left_col,create_links*)
## CONSOLIDATE?(*source_refs,function_num, top_row,left_col,create_links*)

Equivalent to the Data Consolidate command. Consolidates data from multiple ranges on multiple worksheets into a single range on a single worksheet. For detailed information on the Data Consolidate command, see online Help.

***Source_refs*** are quoted external references to areas on other worksheets that contain data to be consolidated on the destination worksheet. *Source_refs* must include the full pathname of the file and the cell reference or named ranges on the worksheet to be consolidated. *Source_refs* must be given as an array, for example:

```
{"Sheet1!IncomeOne","Sheet2!IncomeTwo"}
```

To add or delete *source_refs* from an existing consolidation on a worksheet, reuse the CONSOLIDATE function, specifying the new *source_refs*.

***Function_num*** is a number from 1 to 11 that specifies one of the 11 functions you can use to consolidate data. If *function_num* is omitted, the SUM function, number 9, is used. The functions and their corresponding numbers are:

| *Function_num* | Function |
|---|---|
| 1 | AVERAGE |
| 2 | COUNT |
| 3 | COUNTA |
| 4 | MAX |
| 5 | MIN |
| 6 | PRODUCT |
| 7 | STDEV |
| 8 | STDEVP |
| 9 | SUM |
| 10 | VAR |
| 11 | VARP |

The following arguments correspond to text boxes and check boxes in the Data Consolidate dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Top_row*** corresponds to the Top Row check box. The default is FALSE.

***Left_col*** corresponds to the Left Column check box. The default is FALSE.

If *top_row* and *left_col* are both FALSE or omitted, the data is consolidated by position.

***Create_links*** corresponds to the Create Links To Source Data check box.

#### Tip

Before using the CONSOLIDATE function, use the SELECT function to select a range at least as large as the total number of rows and columns that will be consolidated. For example, if you are consolidating four worksheets, each with four different

categories in the left column, you'll need at least 16 rows on your destination worksheet to accommodate the consolidated data.

*Notes*

- If you use the CONSOLIDATE function with no arguments and there is a consolidation on the active worksheet, Microsoft Excel reconsolidates, using the sources, function, and position attributes used to create the existing consolidation.
- If you use the CONSOLIDATE function with no arguments and there is no consolidation on the active worksheet, the function returns an error.

For more information about consolidation, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*.

*Examples*

The following macro formula reconsolidates an existing consolidation on the active worksheet:

```
CONSOLIDATE()
```

If there is no consolidation on the active worksheet, a message is displayed and the function returns the #VALUE! error value.

In Microsoft Excel for Windows or OS/2, the following macro formula consolidates data from the range $A$10:$F$25 on the worksheets SALES.XLS and SALESWES.XLS using the SUM function and the categories in both the top row and left column:

```
CONSOLIDATE({"C:\EXCEL\SALES.XLS!R10C1:R25C6",
"C:\EXCEL\SALESWES.XLS!R10C1:R25C6"},9,
TRUE,TRUE)
```

In Microsoft Excel for the Macintosh, the following macro formula consolidates data from the ranges named Income on the worksheets named Sales West and Sales East using the COUNTA function and the categories in the top row only:

```
CONSOLIDATE({"Hard Drive:Excel:Financials:Sales
West'!Income","Hard Drive:Excel:Financials:Sales
East'!Income"},3,TRUE,FALSE)
```

*Related Functions*

| | |
|---|---|
| CHANGE.LINK | Changes supporting worksheet links |
| LINKS | Returns the names of all linked documents |
| OPEN.LINKS | Opens specified supporting documents |
| UPDATE.LINK | Updates a link |

*Macro Sheets Only*

## COPY()

Equivalent to the Edit Copy command. Defines the selection that will be copied when you use the PASTE or PASTE.SPECIAL function. For detailed information on the Edit Copy command, see online Help.

*Macro Sheets Only*

## COPY.CHART(*size_num*)

Equivalent to the Edit Copy Chart command in Microsoft Excel for the Macintosh version 1.5 or earlier.

*Note*

This function is included for compatibility with macros written with Microsoft Excel for the Macintosh version 1.5 or earlier. You can copy a chart with the COPY.PICTURE function by omitting the *appearance* argument.

***Size_num*** is the number 1 or 2 specifying how you want the chart copied.

| *Size_num* | Description |
|---|---|
| 1 | As shown on screen |
| 2 | As shown when printed |

*Related Function*

| | |
|---|---|
| COPY.PICTURE | Creates a picture of the current selection |



*Macro Sheets Only*

## COPY.PICTURE(*appearance_num,size_num*)

Equivalent to the Edit Copy Picture command. This command appears on the Edit menu if you hold down SHIFT while choosing the menu. Copies the selection to the Clipboard as a graphic. Use COPY.PICTURE to create an image of the current selection or chart for use in another program. For detailed information on the Edit Copy Picture command, see online Help.

*Note*

Graphics are created differently on screen and on a printer. Thus, the printed picture may look different from the one on screen.

***Appearance_num*** is a number describing how to copy the picture.

| *Appearance_num* | Action |
|---|---|
| 1 | Copies a picture as close as possible to the picture displayed on your screen |
| 2 | Copies what you would see if you printed the selection |

***Size_num*** is a number describing how to copy the picture and only applies if the current selection is a chart.

| *Size_num* | Action |
|---|---|
| 1 | Copies the picture in the same size as the window on which the chart is displayed |
| 2 | Copies what you would see if you printed the chart |

*Examples*

The following macro formula copies part of a worksheet as shown on the screen:

```
COPY.PICTURE(1)
```

The following macro formula copies a chart as it appears on the screen, in the printed size:

```
COPY.PICTURE(1,2)
```

*Related Functions*

| | |
|---|---|
| COPY | Copies information to the Clipboard |
| CUT | Defines the selection that will be moved when you use PASTE |
| PASTE | Pastes copied data |
| PASTE.PICTURE.LINK | Pastes a linked picture of the currently copied area |

## COS(*number*)

Returns the cosine of the given angle.

***Number*** is the angle in radians for which you want the cosine. If the angle is in degrees, multiply it by PI()/180 to convert it to radians.

*Examples*

COS(1.047) equals 0.500171.

COS(60*PI()/180) equals 0.5, the sine of 60 degrees

*Related Functions*

| | |
|---|---|
| ACOS | Returns the arccosine of a number |
| COSH | Returns the hyperbolic cosine of a number |
| PI | Returns the value π |

## COSH(*number*)

Returns the hyperbolic cosine of *number*.

The formula for the hyperbolic cosine is:

$$\text{COSH}(z) = \frac{e^z + e^{-z}}{2}$$

*Examples*

COSH(4) equals 27.30823284

COSH(EXP(1)) equals 7.610125139, where EXP(1) is e, the base of the natural logarithm

### Related Functions

| | |
|---|---|
| ACOSH | Returns the inverse hyperbolic cosine of a number |
| SINH | Returns the hyperbolic sine of a number |
| TANH | Returns the hyperbolic tangent of a number |

## COUNT(value1,value2,...)

Counts how many numbers are in the list of arguments. Use COUNT to get the number of entries in a number field in a range or array of numbers.

***Value1, value2,...*** are 1 to 14 arguments that can contain or refer to a variety of data types, but only numbers are counted.

- Arguments that are numbers, null, logical values, dates, or text representations of numbers are counted; arguments that are error values or text that cannot be translated into numbers are ignored.
- If an argument is an array or reference, only numbers in that array or reference are counted. Empty cells, logical values, text, or error values in the array or reference are ignored.

### Examples

| | A | B | C | D |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | Sales | | | |
| 4 | 12/8/90 | | | |
| 5 | | | | |
| 6 | 19 | | | |
| 7 | 22.34 | | | |
| 8 | | | | |
| 9 | #DIV/0! | | | |

In the preceding worksheet:
COUNT(A6:A7) equals 2
COUNT(A4:A7) equals 3
COUNT(A2,A6:A9,"Twelve",5) equals 3
COUNT(A1:A9,,"2"), where "2" is a text representation of a number, equals 5
COUNT(0.1,TRUE,"three",4,,6.6666,700,,9,#DIV/0!) equals 8

### Related Functions

| | |
|---|---|
| AVERAGE | Returns the average of its arguments |
| COUNTA | Counts how many values are in the list of arguments |
| DCOUNT | Counts the cells containing numbers from a specified database and criteria |
| DCOUNTA | Counts nonblank cells from a specified database and criteria |
| SUM | Adds its arguments |

## COUNTA(value1,value2,...)

Counts the number of nonblank values in the list of arguments. Use COUNTA to count the number of cells with data in a range or array.

***Value1, value2,...*** are 1 to 14 arguments representing the values you want to count. In this case, a value is any type of information, including empty text ("") but not including empty cells. If an argument is an array or reference, empty cells within the array or reference are ignored.

### Examples

| | A | B | C | D |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | Sales | | | |
| 4 | 12/8/90 | | | |
| 5 | | | | |
| 6 | 19 | | | |
| 7 | 22.34 | | | |
| 8 | | | | |
| 9 | #DIV/0! | | | |



In the preceding worksheet:
COUNTA(A6:A7) equals 2
COUNTA(A4:A7) equals 3
COUNTA(A3,A6:A9) equals 4
COUNTA(A1:A9) equals 5
COUNTA(1,,1) equals 3
COUNTA(A4:A7,10) equals 4

### Related Functions

| | |
|---|---|
| AVERAGE | Returns the average of its arguments |
| COUNT | Counts how many numbers are in the list of arguments |
| DCOUNT | Counts the cells containing numbers from a specified database and criteria |
| DCOUNTA | Counts nonblank cells from a specified database and criteria |
| PRODUCT | Multiplies its arguments |
| SUM | Adds its arguments |

*Macro Sheets Only*

## CREATE.NAMES(top,left,bottom,right)
## CREATE.NAMES?(top,left,bottom,right)

Equivalent to the Formula Create Names command (full menus). Use CREATE.NAMES to quickly create names from text labels on a worksheet. For detailed information on the Formula Create Names command, see online Help.

Arguments are logical values corresponding to check boxes in the Formula Create Names dialog box. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE or omitted, Microsoft Excel turns off the check box.

***Top*** corresponds to the Top Row check box.

***Left*** corresponds to the Left Column check box.

***Bottom*** corresponds to the Bottom Row check box.

***Right*** corresponds to the Right Column check box.

### Note

The cell containing the label text that Microsoft Excel uses to create the names is not included in the resulting named range.

### Examples

| | A | B | C | D |
|---|---|---|---|---|
| 1 | | Jan | Feb | Mar |
| 2 | Apples | | | |
| 3 | Oranges | | | |
| 4 | Bananas | | | |

In the worksheet above, when A1:D4 is selected, CREATE.NAMES(TRUE,TRUE) creates the following named ranges:

| | |
|---|---|
| Jan | $B$2:$B$4 |
| Feb | $C$2:$C$4 |
| Mar | $D$2:$D$4 |
| Apples | $B$2:$D$2 |
| Oranges | $B$3:$D$3 |
| Bananas | $B$4:$D$4 |

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | | Jan | Feb | Mar | |
| 2 | Apples | | | | Macintosh |
| 3 | Oranges | | | | Navel |
| 4 | Bananas | | | | Brazilian |

In the worksheet above, when A1:E4 is selected, CREATE.NAMES(TRUE,TRUE,,TRUE) creates the same names as the previous example, plus:

| | |
|---|---|
| Macintosh | $B$2:$D$2 |
| Navel | $B$3:$D$3 |
| Brazilian | $B$4:$D$4 |

### Related Functions

| | |
|---|---|
| APPLY.NAMES | Replaces references and values with their corresponding names |
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| DELETE.NAME | Deletes a name |
| FORMULA.GOTO | Selects the specified reference or named range |

Macro Sheets Only

## CREATE.OBJECT(object_type,ref_1, x_offset1,y_offset1,ref_2,x_offset2,y_offset2)

**Form 1: For lines, rectangles, ovals, arcs, and pictures**

## CREATE.OBJECT(object_type,ref_1, x_offset1,y_offset1,ref_2,x_offset2,y_offset2,text)

**Form 2: For text boxes and buttons**

## CREATE.OBJECT(object_type,ref_1, x_offset1,y_offset1,ref_2,x_offset2,y_offset2,xy_series)

**Form 3: For embedded charts**

Draws an object on a worksheet or macro sheet and returns a value identifying the object created.

***Object_type*** is a number from 1 to 8 specifying the type of object to create.

| *Object_type* | Object |
|---|---|
| 1 | Line |
| 2 | Rectangle |
| 3 | Oval |
| 4 | Arc |
| 5 | Embedded chart |
| 6 | Text box |
| 7 | Button |
| 8 | Picture (created with the camera tool) |

***Ref_1*** is the cell from which the upper-left corner of the object is drawn.

***X_offset1*** is the horizontal distance from the upper-left corner of *ref_1* to the upper-left corner of the object. *X_offset1* is measured in points. A point is 1/72nd of an inch.

***Y_offset1*** is the vertical distance from the upper-left corner of *ref_1* to the upper-left corner of the object. The position is specified in points.

***Ref_2*** is the cell from which the lower-right corner of the object is drawn. *Ref_2* can be omitted when you are creating a picture.

***X_offset2*** is the horizontal distance from the upper-left corner of *ref_2* to the lower-right corner of the object. The position is specified in points.

***Y_offset2*** is the vertical distance from the upper-left corner of *ref_2* to the lower-right corner of the object. The position is specified in points.

***Text*** specifies the text that appears in a text box or button. If *text* is omitted for a button, the button is named "Button *n*", where *n* is a number. If *object_type* is not 6 or 7, *text* is ignored.

***Xy_series*** is a number from 0 to 3 that specifies how data is arranged in a chart and corresponds to options in the First Row/Column Contains dialog box. *Xy_series* is ignored unless *object_type* is 5 (chart).

| *Xy_series* | Result |
|---|---|
| 0 | Displays dialog box if selection is ambiguous. |
| 1 or omitted | First row/column is the first data series. |
| 2 | First row/column are the category (x) axis labels. |
| 3 | First row/column are x-values; created chart is an xy (scatter) chart. |

### Notes

- CREATE.OBJECT returns the object identifier of the object it created. Object identifiers include text describing the object, such as "Text" or "Oval," and a number indicating the order in which the object was created. For example, CREATE.OBJECT returns "Oval 3" after creating an oval that is the third object on the document.
- If the offsets are not specified, the object is drawn from the upper-left corner of *ref_1* to the upper-left corner of *ref_2*.
- If the object is not a picture and either *ref_1* or *ref_2* is omitted, CREATE.OBJECT returns the #VALUE! error value and does not create the object.



- If a chart is active, CREATE.OBJECT returns the #VALUE! error value.
- CREATE.OBJECT also selects the object.
- You must use the COPY function before the CREATE.OBJECT function to create a chart or a picture. See the last example.

### Tip

To assign a macro to an object, use the ASSIGN.TO.OBJECT function immediately after creating the object.

### Examples

The following macro formula creates an oval between cells $C$18 and $D$50 with the specified offsets:

```
CREATE.OBJECT(3,$C$18,20,30,$D$50,20,30)
```

The following macro formula creates a text box between cells $B$10 and $H$20 and inserts the specified text:

```
CREATE.OBJECT(6,$B$10,,,$H$20,,,"Insert the quarterly sales figures in this quadrant.")
```

The following macro formula creates a button between cells $B$10 and $C$15 with the text Run Report:

```
CREATE.OBJECT(7,$B$10,,,$C$15,,,"Run Report")
```

The following two macros create sheet pictures and worksheet charts, respectively. You must use the COPY function before the CREATE.OBJECT function to create charts or pictures.

| | B | C |
|---|---|---|
| 1 | *Formulas* | *Comments* |
| 2 | | |
| 3 | **CreateWorksheetPicture** | |
| 4 | =COPY() | Copies area for picture |
| 5 | =CREATE.OBJECT(8,"R19C2",20,160,"R26C6",60,180) | Creates the sheet picture |
| 6 | =CANCEL.COPY() | Cancels the copy marquee |
| 7 | =RETURN() | |
| 8 | | |
| 9 | **CreateWorksheetChart** | |
| 10 | =COPY() | Copies values for chart |
| 11 | =CREATE.OBJECT(5,"R1C1",580,60,"R11C6",20,100) | Creates worksheet chart |
| 12 | =CANCEL.COPY() | Cancels the copy marquee |
| 13 | =RETURN() | |

### Related Functions

| | |
|---|---|
| ASSIGN.TO.OBJECT | Assigns a macro to an object |
| FORMAT.MOVE | Moves the selected object |
| FORMAT.SIZE | Changes the size of the selected object |
| GET.OBJECT | Returns information about an object |
| PLACEMENT | Determines an object's relationship to underlying cells |

Macro Sheets Only

## CREATE.PUBLISHER(file_text, appearance_num,size_num)

Equivalent to the Edit Create Publisher command. Publishes the selected range or chart to an edition file for use by other Macintosh applications. For detailed information on the Edit Create Publisher command, see online Help.

### Note

This function is only available if you are using Microsoft Excel for the Macintosh with system software version 7.0 or later.

***File_text*** is a quoted text string to be used as the name of the new file that will contain the selected data. If *file_text* is omitted, Microsoft Excel uses the format "DocumentName Edition *n*", where DocumentName is the name of the document from which the publisher is being created, Edition indicates that the file is an edition file, and *n* is a unique integer.

For example, if you omit *file_text* and are publishing a selection from a worksheet named Seasonal, and it is your third publisher from that document in the current work session, the default name of the publisher would be "Seasonal Edition 3".

***Appearance_num*** specifies whether the selection is to be published as shown on screen or as shown when printed. The default value for *appearance_num* is 1 if the selection is a sheet and 2 if the selection is a chart.

| *Appearance_num* | Selection is published |
|---|---|
| 1 | As shown on screen |
| 2 | As shown when printed |

***Size_num*** specifies the size at which to publish a chart. *Size_num* is available only if a chart is to be published.

| *Size_num* | Chart size is published |
|---|---|
| 1 or omitted | As shown on screen |
| 2 | As shown when printed |

### Examples

The following macro formula publishes the current selection, as shown on screen, to a file called March Data Edition 1:

```
CREATE.PUBLISHER("March Data Edition 1",1)
```

The following macro formula publishes the active chart to an edition file called March Sales Chart, with the appearance and size as shown on screen:

```
CREATE.PUBLISHER("March Sales Chart",1,1)
```

### Related Functions

| | |
|---|---|
| EDITION.OPTIONS | Sets publisher and subscriber options |
| GET.LINK.INFO | Returns information about a link |
| SUBSCRIBE.TO | Inserts contents of an edition into the active document |
| UPDATE.LINK | Updates a link to another document |

Macro Sheets Only

## CUSTOM.REPEAT(*macro_text,repeat_text, record_text*)

Creates an Edit Repeat command for custom commands and allows custom commands to be recorded using the macro recorder.

***Macro_text*** is a reference to a macro you want to run when the Edit Repeat command is chosen. *Macro_text* can be the name or cell reference of a macro. If *macro_text* is omitted, a repeat is not set but a formula can still be recorded when Edit Repeat is chosen.

***Repeat_text*** is the text you want to use as the Edit Repeat command.

***Record_text*** is the formula you want to have recorded. For example, if the user chooses Edit Repeat after choosing a command named Run Reports in Macro 1, the *record_text* argument would be "=Macro1!RunReports()", where RunReports is the name of the macro called by the Run Reports command. References in *record_text* must be in R1C1 format.

### Tip

Use CUSTOM.REPEAT directly after the macro functions you want to be able to repeat, because other macro functions following CUSTOM.REPEAT might nullify its effect.

### Example

See CUSTOM.UNDO.

Macro Sheets Only

## CUSTOM.UNDO(*macro_text,undo_text*)

Creates an Edit Undo command for custom commands.

***Macro_text*** is a reference to the macro you want to run when Edit Undo is chosen. *Macro_text* can be the name or cell reference of a macro.

***Undo_text*** is the text you want to use as the Edit Undo command.

### Example

The following macro creates a command when Edit Undo is chosen which undoes the multiplication of the contents of the active cell.



| | B | C |
|---|---|---|
| 1 | *Formulas* | *Comments* |
| 2 | | |
| 3 | Times100 | |
| 4 | =SET.NAME("Mcell",ACTIVE.CELL()) | Names active cell "Mcell" |
| 5 | =FORMULA.GOTO(Mcell) | Activates Mcell |
| 6 | =FORMULA(Mcell*100) | Mcell * 100 |
| 7 | =CUSTOM.UNDO("UndoMult","&Undo Times100") | Sets custom undo |
| 8 | =CUSTOM.REPEAT("Times100","&Repeat Times100") | Sets custom repeat |
| 9 | =RETURN() | |
| 10 | | |
| 11 | UndoMult | |
| 12 | =FORMULA.GOTO(Mcell) | Activates Mcell |
| 13 | =FORMULA(Mcell/100) | Divides Mcell by 100 |
| 14 | =CUSTOM.UNDO("R5C2","Redo (&u) Times100") | Sets custom redo |
| 15 | =RETURN() | |

### Tip

Use CUSTOM.UNDO directly after the macro functions you want to be able to repeat, because other macro functions following CUSTOM.UNDO might nullify its effect.

Macro Sheets Only

## CUT()

Equivalent to the Edit Cut command. Deletes a selection that will be moved when you use the PASTE function. For detailed information on the Edit Cut command, see online Help.

Macro Sheets Only

## DATA.DELETE()
## DATA.DELETE?()

Equivalent to the Data Delete command (full menus). Deletes data that matches the current criteria in the current database. For detailed information on the Data Delete command, see online Help.

In the dialog-box form, DATA.DELETE?, Microsoft Excel displays a message warning you that matching records will be permanently deleted, and you can approve or cancel. In the plain form, DATA.DELETE, matching records are deleted without any message being displayed.

Macro Sheets Only

## DATA.FIND(*logical*)

Equivalent to the Data Find and Data Exit Find commands. Selects records in the database range which match criteria in the criteria range. For detailed information on the Data Find command, see online Help.

***Logical*** is a logical value that specifies whether to enter or exit the Data Find mode. If logical is TRUE, Microsoft Excel carries out the Data Find command; if FALSE, Microsoft Excel carries out the Data Exit Find command.

### Related Functions

| | |
|---|---|
| DATA.FIND.NEXT | Finds next matching record in a database |
| DATA.FIND.PREV | Finds previous matching record in a database |

Macro Sheets Only

## DATA.FIND.NEXT()
## DATA.FIND.PREV()

Equivalent to pressing the DOWN ARROW or UP ARROW key after the Data Find command has been chosen. Finds the next or previous matching record in a database. If the function cannot find a matching record, it returns the logical value FALSE. For detailed information on the Data Find command, see online Help.

### Related Function

| | |
|---|---|
| DATA.FIND | Enters or exits Data Find mode |

Macro Sheets Only

## DATA.FORM()

Equivalent to the Data Form command. Displays the data form. For detailed information on the Data Form command, see online Help.

If no database is defined on the active sheet, DATA.FORM returns the #VALUE! error value and interrupts the macro.

### Note

You can create custom data forms by using the Dialog Editor to design a data entry dialog box and then naming the dialog-box definition table "Data_form" on the worksheet containing the database you want to use with the data form. You do not need to include OK and Close buttons in your custom data form, because the data form includes its own buttons.

For more information about data forms, see Chapter 9, "Creating and Using a Database on a Worksheet," in the *Microsoft Excel User's Guide*.

---

*Macro Sheets Only*

## DATA.SERIES(*rowcol,type_num,date_num, step_value,stop_value*)

## DATA.SERIES?(*rowcol,type_num,date_num, step_value,stop_value*)

Equivalent to the Data Series command (full menus). Use DATA.SERIES to enter an incrementally increasing or decreasing series of values or dates on a worksheet or macro sheet. For detailed information on the Data Series command, see online Help.

***Rowcol*** is a number that specifies where the series should be entered.

| *Rowcol* | Enter series in |
|---|---|
| 1 | Rows |
| 2 | Columns |

***Type_num*** is a number from 1 to 3 that specifies the type of series.

| *Type_num* | Type of series |
|---|---|
| 1 | Linear |
| 2 | Growth |
| 3 | Date |

***Date_num*** is a number from 1 to 4 that specifies the date unit of the series, as shown below. To use the *date_num* argument, the *type_num* argument must be 3. If *type_num* is 3 and no *date_num* argument is specified, *date_num* is assumed to be 1.

| *Date_num* | Date unit |
|---|---|
| 1 | Day |
| 2 | Weekday |
| 3 | Month |
| 4 | Year |

***Step_value*** is a number that specifies the step value for the series. If *step_value* is omitted, 1 is assumed.

***Stop_value*** is a number that specifies the stop value for the series. If *stop_value* is omitted, DATA.SERIES continues filling the series until the end of the selected range.

### Note

If you specify a value for *stop_value* that is lower than the value in the active cell of the selection, DATA.SERIES takes no action.

---

## DATE(*year,month,day*)

Returns the serial number of a particular date. For more information about serial numbers, see NOW.

***Year*** is a number from 1900 to 2078. To specify a year in the range 1920 to 2019, you can give the last two digits of the year. To specify a year before 1920 or after 2019, give all four digits of the year.

***Month*** is a number representing the month of the year. If *month* is greater than 12, then *month* adds that number of months to the first month in the year specified. For example, DATE(90,14,2) returns the serial number representing February 2, 1991.

***Day*** is a number representing the day of the month. If *day* is greater than the number of days in the month specified, then *day* adds that number of days to the first day in the month. For example, DATE(91,1,35) returns the serial number representing February 4, 1991.



### Note

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

### Tip

The DATE function is most useful in formulas where *year*, *month*, and *day* are formulas, not constants.

### Examples

Using the 1900 Date System (the default in Microsoft Excel for Windows or OS/2), DATE(91,1,1) equals 33239, the serial number corresponding to January 1, 1991.

Using the 1904 Date System (the default in Microsoft Excel for the Macintosh), DATE(91,1,1) equals 31777, the serial number corresponding to January 1, 1991.

The following example uses the 1900 Date System.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | Date of Sales | | | |
| 2 | Month | Day | Year | Term (in days) |
| 3 | 1 | 24 | 91 | 90 |
| 4 | | | | |

In the preceding worksheet, to find the due date for a bill due 90 days from 1/24/91, use: DATE(C3,A3,B3+D3), which equals 33352. The date corresponding to serial number 33352 is April 24, 1991.

### Related Functions

| | |
|---|---|
| DATEVALUE | Converts a date in the form of text to a serial number |
| DAY, MONTH, and YEAR | Convert serial numbers to days, months, and years |
| NOW | Returns the serial number of the current date and time |
| TIMEVALUE | Converts a time in the form of text to a serial number |
| TODAY | Returns the serial number of today's date |

---

## DATEVALUE(*date_text*)

Returns the serial number of the date represented by *date_text*. Use DATEVALUE to convert a date represented by text to a serial number.

### Note

Most functions automatically convert date values to serial numbers.

***Date_text*** is quoted text that returns a date in a Microsoft Excel date format. Using the default date system in Microsoft Excel for Windows or OS/2, *date_text* must represent a date from January 1, 1900 to December 31, 2078. Using the default date system in Microsoft Excel for the Macintosh, *date_text* must represent a date from January 1, 1904 to December 31, 2078. DATEVALUE returns the #VALUE! error value if *date_text* is out of this range.

If the year portion of *date_text* is omitted, DATEVALUE uses the current year from your computer's built-in clock. Time information in *date_text* is ignored.

### Note

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

### Examples

The following examples use the 1900 Date System:

DATEVALUE("8/22/55") equals 20323

DATEVALUE("22-AUG-55") equals 20323

Assuming your computer's built-in clock is set to 1991 and you are using the 1900 Date System:

DATEVALUE("5-JUL") equals 33424

### Related Functions

| | |
|---|---|
| NOW | Returns the serial number of the current date and time |
| TIMEVALUE | Converts a time in the form of text to a serial number |
| TODAY | Returns the serial number of today's date |

## DAVERAGE(database,field,criteria)

Averages the values in the *field* column of records in the *database* which match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

## DAY(serial_number)

Returns the day of the month corresponding to *serial_number*. The day is given as an integer ranging from 1 to 31.

**Serial_number** is the date-time code used by Microsoft Excel for date and time calculations. You can give *serial_number* as text, such as "4-15-91" or "15-Apr-1991", instead of as a number. The text is automatically converted to a serial number. For more information about *serial_number*, see NOW.

### Note

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

### Examples

DAY("4-Jan") equals 4

DAY("15-Apr-1991") equals 15

DAY("8/11/91") equals 11

### Related Functions

| | |
|---|---|
| NOW | Returns the serial number of the current date and time |
| TODAY | Returns the serial number of today's date |
| YEAR, MONTH, WEEKDAY, HOUR, MINUTE, and SECOND | Convert serial numbers into years, months, days of the week, hours, minutes, and seconds |

## DAYS360(start_date,end_date)

Returns the number of days between two dates based on a 360-day year (twelve 30-day months). Use this function to help compute payments if your accounting system is based on twelve 30-day months.

**Start_date** and **end_date** are the two dates between which you want to know the number of days.

- The arguments can be either quoted text strings using numbers to represent the month, day, and year (for example, "1/30/91" or "1-30-91"), or they can be serial numbers representing the dates.
- If *start_date* occurs after *end_date*, DAYS360 returns a negative number.

### Note

To determine the number of days between two dates in a normal year, you can use normal subtraction—for example, "12/31/91"–"1/1/91" equals 364.

### Example

DAYS360("1/30/91","2/1/91") equals 1

### Related Function

| | |
|---|---|
| DAY | Converts a serial number to a day of the month |



## DCOUNT(database,field,criteria)

Counts the cells that contain numbers that match the *criteria* in the *field* column of records in the *database*.

The *field* argument is optional. If *field* is omitted, DCOUNT counts all records in the database which match the criteria. For more information on Microsoft Excel database functions, see "D*function*."

## DCOUNTA(database,field,criteria)

Counts the cells that are not blank and that satisfy the *criteria* in the *field* column of records in the *database*.

## DDB(cost,salvage,life,period,factor)

Returns the depreciation of an asset for a specific period using the double-declining balance method or some other method you specify.

**Cost** is the initial cost of the asset.

**Salvage** is the value at the end of the depreciation (sometimes called the salvage value of the asset).

**Life** is the number of periods over which the asset is being depreciated (sometimes called the useful life of the asset).

**Period** is the period for which you want to calculate the depreciation. *Period* must use the same units as *life*.

**Factor** is the rate at which the balance declines. If *factor* is omitted, it is assumed to be 2 (the double-declining balance method).

All five arguments must be positive numbers.

### Note

The double-declining balance method computes depreciation at an accelerated rate. Depreciation is highest in the first period and decreases in successive periods. DDB uses the following formula to calculate depreciation for a period:

((*cost* – total depreciation from prior periods) * *factor*) / *life*

Change *factor* if you do not want to use the double-declining balance method.

### Examples

Suppose a factory purchases a new machine. The machine costs $2,400 and has a lifetime of 10 years. The salvage value of the machine is $300. The following examples show depreciation over several periods. The results are rounded to two decimal places.

DDB(2400,300,3650,1) equals $1.32, the first day's depreciation. Microsoft Excel automatically assumes that *factor* is 2.

DDB(2400,300,120,1,2) equals $40.00, the first month's depreciation.

DDB(2400,300,10,1,2) equals $480.00, the first year's depreciation.

DDB(2400,300,10,2,1.5) equals $306.00, the second year's depreciation using a *factor* of 1.5 instead of the double-declining balance method.

DDB(2400,300,10,10) equals $22.12, the 10th year's depreciation. Microsoft Excel automatically assumes that *factor* is 2.

### Related Functions

| | |
|---|---|
| SLN | Returns the straight-line depreciation of an asset for one period |
| SYD | Returns the sum-of-years' digits depreciation of an asset for a specified period |
| VDB | Returns the depreciation of an asset for a partial period |

Macro Sheets Only

## DEFINE.NAME(*name_text,refers_to,macro_type, shortcut_text,hidden*)

## DEFINE.NAME?(*name_text,refers_to, macro_type,shortcut_text*)

Equivalent to the Formula Define Name command. Defines a name on the active worksheet or macro sheet. For detailed information on the Formula Define Name command, see online Help.

***Name_text*** is the text you want to use as the name. Names must start with a letter, cannot include spaces or symbols, and cannot look like cell references.

***Refers_to*** describes what *name_text* should refer to, and can be any of the following values:

| If *refers_to* is | Then *name_text* is |
|---|---|
| A number, text, or logical value | Defined to refer to that value |
| An external reference, such as !$A$1 or SALES!$A$1:$C$3 | Defined to refer to those cells |
| A formula in the form of text, such as "=2*PI()/360". (If the formula contains references, they must be R1C1-style references, such as "=R2C2*(1+RC[–1])".) | Defined to refer to that formula |
| Omitted | Defined to refer to the current selection |

The next two arguments, *macro_type* and *shortcut_text*, apply only if the document in the active window is a macro sheet.

***Macro_type*** is a number from 1 to 3 that indicates the type of macro.

| *Macro_type* | Type of macro |
|---|---|
| 1 | Function macro (also known as a custom function) |
| 2 | Command macro |
| 3 | None (that is, *name_text* does not refer to a macro) |

If *macro_type* is omitted, it is assumed to be 3.

***Shortcut_text*** is a text value that specifies the macro shortcut key. *Shortcut_text* must be a single letter, such as "z" or "Z".

***Hidden*** is a logical value specifying whether to define the name as a hidden name. If *hidden* is TRUE, Microsoft Excel defines the name as a hidden name; if *hidden* is FALSE or omitted, Microsoft Excel defines the name normally.

For information on using text, number, and logical arguments, see "Converting Data Types" earlier in this book.

### Notes

- You can use hidden names to define values that you want to prevent the user from seeing or changing. Hidden names can only be created with the DEFINE.NAME macro function.
- If you are recording a macro and you define a name to refer to a formula, Microsoft Excel converts A1-style references to R1C1-style references. For example, if the active cell is C2, and you define the name Previous to refer to =B2, Microsoft Excel records that command as DEFINE.NAME("Previous","=RC[–1]").
- In DEFINE.NAME?, the dialog-box form of the function, if *refers_to* is not specified, the current selection is proposed in the Refers To box. Also, if a name is not specified, text in the active cell is proposed as the name.

### Examples

The following formula defines the name Inflation on the active worksheet to refer to the value 0.12:

```
DEFINE.NAME("Inflation",0.12)
```



If the selection is A1:C1, each of the following formulas defines the name Sales on the active worksheet to refer to cells A1:C3:

```
DEFINE.NAME("Sales")
DEFINE.NAME("Sales",SELECTION())
DEFINE.NAME("Sales","=R1C1:R3C3")
```

The following macro formula defines the name Previous to refer to the relative reference RC[–1]:

```
DEFINE.NAME("Previous","=RC[–1]")
```

If the range A1:C3 is selected, then the following macro formula defines the name Sales on the active worksheet to refer to A1:C1, the first row of the selection:

```
DEFINE.NAME("Sales",INDEX(SELECTION(),1,0))
```

The following macro formula defines the name SalesAverage to refer to the formula AVERAGE(A1:F1):

```
DEFINE.NAME("SalesAverage","=AVERAGE(A1:F1)")
```

### Related Functions

| | |
|---|---|
| DELETE.NAME | Deletes a name |
| GET.DEF | Returns a name matching a definition |
| GET.NAME | Returns the definition of a name |
| NAMES | Returns a horizontal text array of all the names that are defined on a document |
| SET.NAME | Defines a name on a macro sheet |

Macro Sheets Only

## DEFINE.STYLE(*style_text,number,font, alignment,border,pattern,protection*)

## DEFINE.STYLE?(*style_text,number,font, alignment,border,pattern,protection*)

**Form 1: By example**

Equivalent to choosing the Define button in the Format Style dialog box. Creates and changes cell styles. There are seven forms of this function. Use this form of DEFINE.STYLE to define styles based on the format of the active cell. To create a style by specifying number, font, and other formats, use forms 2 through 7 of DEFINE.STYLE. For detailed information on the Format Style command, see online Help.

***Style_text*** is the name, as quoted text, that you want to assign to the style.

The following arguments are logical values corresponding to check boxes in the expanded Format Style dialog box. If an argument is TRUE, Microsoft Excel turns on the check box and uses the corresponding format of the active cell in the style; if FALSE, Microsoft Excel turns off the check box and omits formatting descriptions for that attribute. If *style_text* is omitted and all selected cells have identical formatting, the default is TRUE; if cells have different formatting, the default is FALSE.

***Number*** corresponds to the Number check box.

***Font*** corresponds to the Font check box.

***Alignment*** corresponds to the Alignment check box.

***Border*** corresponds to the Border check box.

***Pattern*** corresponds to the Pattern check box.

***Protection*** corresponds to the Protection check box.

### Examples

The following macro formula defines a new style named Headings, which has all the formatting attributes of the active cell.

```
DEFINE.STYLE("Headings")
```

If a Headings style is already defined, the above formula replaces the current definition of the style with the formatting attributes of the active cell.

The following macro formula uses the number, font, and border attributes of the active cell to create a new style called Titles:

```
DEFINE.STYLE("Titles",TRUE,TRUE,FALSE,TRUE,FALSE,
FALSE)
```

### Related Functions

| | |
|---|---|
| APPLY.STYLE | Applies a style to the selection |
| DELETE.STYLE | Deletes a cell style |
| MERGE.STYLES | Imports styles from another document into the active document |

Macro Sheets Only

## DEFINE.STYLE(style_text,attribute_num, format_text)

**Form 2: Number format, using the arguments from the FORMAT.NUMBER function**

## DEFINE.STYLE(style_text,attribute_num, name_text,size_num,bold,italic,underline,strike,color, outline,shadow)

**Form 3: Font format, using the arguments from the FORMAT.FONT function**

## DEFINE.STYLE(style_text,attribute_num, type_num,wrap)

**Form 4: Alignment, using the arguments from the ALIGNMENT function**

## DEFINE.STYLE(style_text,attribute_num, left,right,top,bottom,left_color,right_color,top_color, bottom_color)

**Form 5: Border, using the arguments from the BORDER function**

## DEFINE.STYLE(style_text,attribute_num, apattern,afore,aback)

**Form 6: Pattern, using the arguments from the cell form of the PATTERNS function**

## DEFINE.STYLE(style_text,attribute_num,locked, hidden)

**Form 7: Cell protection, using the arguments from the CELL.PROTECTION function**

Equivalent to choosing the Define button in the Format Style dialog box. Creates and changes cell styles. Use one of these forms of DEFINE.STYLE to select cell formats for a new style or to alter the formats of an existing style. Use the first form of DEFINE.STYLE to define styles based on the format of the active cell. For detailed information on the Format Style command, see online Help.

***Style_text*** is the name, as quoted text, that you want to assign to the style.

***Attribute_num*** is a number from 2 to 7 that specifies which attribute of the style, such as its font, alignment, or number format, you want to designate with this function.

| *Attribute_num* | Specifies |
|---|---|
| 2 | Number format |
| 3 | Font format |
| 4 | Alignment |
| 5 | Border |
| 6 | Pattern |
| 7 | Cell protection |

### Notes

- The remaining arguments are different for each form and are identical to arguments in the corresponding function. For example, form 2 of DEFINE.STYLE defines the number format of a style and corresponds to the FORMAT.NUMBER function. The exception is form 5, which does not include every argument for BORDER. For details on the values you can use for these arguments, see the description under the corresponding function.
- If you define a style using one of these forms, then any attributes you don't explicitly define are set by example.

### Example

The following portion of a macro creates a new style named Totals, sets alignment to centered, sets a number format, and sets the font to Courier, 14-point bold:

| | B | C |
|---|---|---|
| 10 | =DEFINE.STYLE("Totals",4,3) | Sets alignment to centered |
| 11 | =DEFINE.STYLE("Totals",2,"$#,##0_) ;Red($#,##0)") | Sets number format |
| 12 | =DEFINE.STYLE("Totals",3,"Courier",14,TRUE) | Sets font format |



### Related Functions

| | |
|---|---|
| APPLY.STYLE | Applies a style to the selection |
| DELETE.STYLE | Deletes a cell style |
| MERGE.STYLES | Imports styles from another document into the active document |

Macro Sheets Only

## DELETE.ARROW()

Equivalent to the Chart Delete Arrow command. Deletes the selected arrow. For detailed information on the Chart Delete Arrow command, see online Help.

If the selection is not an arrow or if the active document is not a chart, DELETE.ARROW returns the #VALUE! error value and interrupts the macro.

### Tip

Use the SELECT function (chart form), with the number of the arrow you want to delete (for example, SELECT ("Arrow1")), to select the arrow before using the DELETE.ARROW function.

### Related Function

| | |
|---|---|
| DELETE.OVERLAY | Deletes the overlay on a chart |

Macro Sheets Only

## DELETE.BAR(bar_num)

Deletes a custom menu bar.

***Bar_num*** is the ID number of the custom menu bar you want to delete.

### Tip

Rather than trying to discover the menu bar number of the menu bar you want to delete, use a reference to the ADD.BAR function that created the bar. For example, the following macro formula deletes the menu bar created by the ADD.BAR function in the cell named ReportsBar:

```
DELETE.BAR(ReportsBar)
```

For information on custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Related Functions

| | |
|---|---|
| ADD.BAR | Creates a new menu bar |
| SHOW.BAR | Displays a menu bar |

Macro Sheets Only

## DELETE.COMMAND(bar_num,menu, command)

Deletes a command from a custom or built-in menu. Use DELETE.COMMAND to remove commands you don't want the user to have access to.

***Bar_num*** is the menu bar from which you want to delete the command. *Bar_num* can be the ID number of a built-in or custom menu bar. See ADD.COMMAND for a list of ID numbers for built-in menu bars.

***Menu*** is the menu from which you want to delete the command. *Menu* can be the name of a menu as text or the number of a menu.

***Command*** is the command you want to delete. *Command* can be the name of the command as text or the number of the command (the first command on a menu is in position 1).

- If the specified command does not exist, DELETE.COMMAND returns the #VALUE! error value.
- After a command is deleted, the *command* number for all commands below that command is decreased by one.
- When you delete a built-in command, DELETE.COMMAND returns a unique ID number for that command. You can use this ID number with ADD.COMMAND to restore the built-in command to the original menu.

For more information on custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Example

The following formula removes the Compile Reports command from the Reports menu on a custom menu bar created by the ADD.BAR function in a cell named Financials.

```
DELETE.COMMAND(Financials,"Reports", "Compile Reports")
```

### Related Functions

| | |
|---|---|
| ADD.COMMAND | Adds a command to a menu |
| CHECK.COMMAND | Adds or deletes a check mark from a command |
| ENABLE.COMMAND | Enables or disables a command |
| RENAME.COMMAND | Renames a command |

Macro Sheets Only

## DELETE.FORMAT(format_text)

Equivalent to deleting the specified format with the Format Number command. Deletes a specified custom number format.

***Format_text*** is the custom format given as a quoted text string, for example, "000-00-0000". If you specify a built-in Microsoft Excel format, DELETE.FORMAT returns the #VALUE! error value.

### Note

When you delete a custom number format, all numbers formatted with that number format are formatted with the General format.

### Example

The following macro gets the custom number format of the active cell and deletes it.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | RemoveFormat | Removes format of active cell |
| 4 | DelFormat | =GET.CELL(7,ACTIVE.CELL()) | Gets format of active cell |
| 5 | | =DELETE.FORMAT(DelFormat) | Deletes the format |
| 6 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| FORMAT.NUMBER | Applies a number format to the selection |
| GET.CELL | Returns information about a cell |

Macro Sheets Only

## DELETE.MENU(bar_num,menu)

Deletes a menu from a menu bar. Use DELETE.MENU to delete menus you have added to menu bars when the supporting macro sheet is closed (using an Auto_Close macro), or any time you want to remove a menu.

***Bar_num*** is the menu bar from which you want to delete the menu. *Bar_num* can be the number of a Microsoft Excel built-in menu bar or the number returned by a previously run ADD.BAR function. For a list of ID numbers for built-in menu bars, see ADD.COMMAND.

***Menu*** is the menu you want to delete. *Menu* can be either the name of a menu as text or the number of a menu. If the specified menu does not exist, DELETE.MENU returns the #VALUE! error value. After a menu is deleted, the *menu* number for each menu to the right of that menu is decreased by 1.

For more information on custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Example

The following formula deletes the Reports menu from the custom menu bar created by the ADD.BAR function in a cell named Financials:

```
DELETE.MENU(Financials,"Reports")
```

### Related Functions

| | |
|---|---|
| ADD.MENU | Adds a menu to a menu bar |
| DELETE.BAR | Deletes a menu bar |
| DELETE.COMMAND | Deletes a command from a menu |
| ENABLE.COMMAND | Enables or disables a command or menu |



Macro Sheets Only

## DELETE.NAME(name_text)

Equivalent to deleting a name on the active document with the Formula Define Name command. Deletes the specified name. For detailed information on the Formula Define Name command, see online Help.

***Name_text*** is a text value specifying the name that you want to delete. For information on using text arguments, see "Converting Data Types," earlier in this book.

### Important

Formulas that use names in their arguments may return incorrect or error values when a name used in the formula is deleted.

### Example

The following macro formula deletes the name Inflation from the active document:

```
DELETE.NAME("Inflation")
```

### Related Functions

| | |
|---|---|
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| GET.NAME | Returns the definition of a name |
| SET.NAME | Defines a name on a macro sheet |

Macro Sheets Only

## DELETE.OVERLAY()

Equivalent to the Chart Delete Overlay command (full menus). Deletes an overlay from a chart. If the chart has no overlay, DELETE.OVERLAY takes no action and returns TRUE. For detailed information on the Chart Delete Overlay command, see online Help.

### Related Functions

All the 2-D GALLERY functions, which create the ten 2-D chart types, include a *delete_overlay* argument. If *delete_overlay* is set to TRUE, the existing overlay is deleted and the new format is applied to the main chart.

Macro Sheets Only

## DELETE.STYLE(style_text)

Equivalent to choosing the Delete button in the expanded Format Style dialog box. Deletes a style from a document. Cells formatted with the deleted style revert to the Normal style.

For detailed information on the Format Style command, see online Help.

***Style_text*** is the name of a style to be deleted. If *style_text* does not exist, DELETE.STYLE returns the #VALUE! error value and interrupts the macro.

For more information on using text arguments, see "Converting Data Types" earlier in this book.

### Note

You can only delete styles from the active document. External references are not permitted as part of the *style_text* argument.

### Examples

The following macro formula deletes the style named Titles on the active document.

```
DELETE.STYLE("Titles")
```

The following macro formula deletes the style of the active cell:

```
DELETE.STYLE(GET.CELL(40))
```

### Related Functions

| | |
|---|---|
| APPLY.STYLE | Applies a style to the selection |
| DEFINE.STYLE | Creates or changes a cell style |
| MERGE.STYLES | Merges styles from another document into the active document |

Macro Sheets Only

## DEMOTE(*rowcol*)
## DEMOTE?(*rowcol*)

Equivalent to clicking the demote button. Demotes the selected rows or columns in an outline. Use DEMOTE to change the configuration of an outline by demoting rows or columns of information.

***Rowcol*** specifies whether to demote rows or columns.

| *Rowcol* | Demotes |
|---|---|
| 1 or omitted | Rows |
| 2 | Columns |

### Notes

- If the selection consists of an entire row or rows, then rows are demoted even if *rowcol* is 2. Similarly, selection of an entire column overrides *rowcol* 1.
- Also, if the selection is unambiguous (an entire row or column), then DEMOTE? will not display the dialog box.

### Related Functions

| | |
|---|---|
| PROMOTE | Promotes the selected rows or columns in an outline |
| SHOW.DETAIL | Expands or collapses a portion of an outline |
| SHOW.LEVELS | Displays a specific number of levels of an outline |

Macro Sheets Only

## DEREF(*reference*)

Returns the value of the cells in a reference.

***Reference*** is the cell or cells from which you want to obtain a value. If *reference* is the reference of a single cell, DEREF returns the value of that cell. If *reference* is the reference of a range of cells, DEREF returns the array of values in those cells. If *reference* refers to the active sheet, it must be an absolute reference. Relative references are converted to absolute references.

### Note

In most formulas, there is no difference between using a value and using the reference of a cell containing that value. The reference is automatically converted to the value, as necessary. For example, if cell A1 contains the value 2, then the formula =A1+1, like the formula =2+1, returns the result 3, because the reference A1 is converted to the value 2. However, in a few functions, such as the SET.NAME function, references are not automatically converted to values. Instead, those functions behave differently depending on whether an argument is a reference or a value.

### Example

For an example of the use of this function, see SET.NAME.

## Dfunction(*database,field,criteria*)

# Database Functions

This section describes the 12 worksheet functions used for Microsoft Excel database calculations. For information on database structure and criteria, see Chapter 10, "Analyzing and Reporting Database Information," in the *Microsoft Excel User's Guide*. Each of these functions, referred to collectively as *Dfunction*, uses three arguments: *database*, *field*, and *criteria*. These arguments refer to the worksheet ranges that are used in the database function. The following illustration shows a sample database and criteria range.



| | A | B | C | D |
|---|---|---|---|---|
| 1 | Name | Species | Age | Value |
| 2 | | | >500 | |
| 3 | Name | Species | Age | Value |
| 4 | Paul | Mallard | 26 | 5 |
| 5 | Wally | Wombat | 2 | 650 |
| 6 | Jo | Sun Bear | 2 | 700 |
| 7 | John | Crow | 22 | 150 |
| 8 | Steve | Carp | 25 | 100 |
| 9 | Lesley | White Tiger | 2 | 1000 |
| 10 | Dayle | Puffin | 15 | 50 |

The *database* is the range A3:D10; the *fields* are Name, Species, Age, and Value; and the *criteria* are listed in cells A1:D2.

To average the ages of all animals that are worth more than $500, you could use the DAVERAGE function as follows:

```
DAVERAGE(A3:D10,"Age",A1:D2)
```

***Database*** is the range of cells that make up the database.

- A Microsoft Excel database is a contiguous range of cells organized into records (rows) and fields (columns). The *database* reference can be entered as a cell range, such as A3:D10 in the example above, or as a name assigned to a range.
- If you use the Data Set Database command on a selected range of cells, Microsoft Excel automatically names the range Database. For example, if you had selected the range A3:D10 in the example above and chosen the Data Set Database command, you could use this formula:

```
DAVERAGE(Database,"Age",A1:D2)
```

### Tips

- You can use any range that includes field headings at the top of each row of data as the *database* argument. This is useful if you want to perform *Dfunction* calculations on more than one range of data on a worksheet.
- Remember that when you use a name as an argument, the name should not be enclosed in double quotation marks. When you use a field heading, you must use double quotation marks.

***Field*** indicates which field is used in the function. Database fields are columns of data with an identifying field name in the first row. The *field* argument can be given as text, such as "Age" or "Value" in the example above, or as a field number: 1 for the first field (Name, in the example above), 2 for the second (Species), and so on.

***Criteria*** is the range of cells that contains the database criteria. The *criteria* reference can be entered as a cell range, such as A1:D2 in the example above, or as a name assigned to a range. If you use the Data Set Criteria command on a selected range of cells, Microsoft Excel automatically names the range Criteria. For example, if you had selected the range A1:D2 in the example above and chosen the Data Set Criteria command, you could use this formula:

```
DAVERAGE(A3:D10,"Age",Criteria)
```

### Tips

- You can use any range for the *criteria* argument, as long as it includes at least one field name and at least one cell for specifying a criteria comparison value.
- For example, if the range G1:G2 contains the field heading Income in G1 and the amount 10,000 in G2, you could define the range as MatchIncome and use that name as the *criteria* argument in your database functions.
- To perform an operation on an entire column in a database, enter a blank line below the field names in the criteria range.

### Examples

The following illustration shows a sample database of employee records. Each row contains information about one employee. The database is defined as A3:E8, and the criteria are defined as A1:E2.

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Name | Hire Date | Age | Sex | Income |
| 2 | | >1/1/84 | | | |
| 3 | Name | Hire Date | Age | Sex | Income |
| 4 | Andrews, Jane | 5/6/79 | 42 | F | $17,000 |
| 5 | Brown, Jessica | 6/7/85 | 21 | F | $39,000 |
| 6 | Miller, Jim | 12/1/70 | 64 | M | $38,000 |
| 7 | Sebring, Dave | 9/23/84 | 34 | M | $18,000 |
| 8 | Smith, Cindy | 2/1/83 | 34 | F | $21,000 |

DAVERAGE(Database,"Income",Criteria) equals $28,500, the average Income for employees hired after 1/1/84 (Jessica Brown and Dave Sebring).

DAVERAGE(Database,3,Criteria) equals 27.5, the average Age for employees hired after 1/1/84.

DCOUNT(Database,"Income",Criteria) equals 2. This function looks at the records of employees hired after 1/1/84 and counts how many of the Income fields in those records contain numbers.

DCOUNT(Database,"Name",Criteria) equals 0. This function looks at the records of employees hired after 1/1/84 and counts how many of the Name fields in those records contain numbers.

DCOUNTA(Database,"Income",Criteria) equals 2. This function looks at the records of employees hired after 1/1/84 and counts how many of the Income fields in those records are not blank.

DMAX(Database,"Income",Database) equals $39,000, the maximum Income value.

DMIN(Database,5,Criteria) equals $18,000, the minimum Income for employees hired after 1/1/84.

The following illustration shows a database for a small orchard. Each record contains information about one tree. The database is defined as A1:E7, and the criteria are defined as A9:F11.

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | Tree | Height | Age | Yield | Profit | |
| 2 | Apple | 18 | 20 | 14 | $105.00 | |
| 3 | Pear | 12 | 12 | 10 | $96.00 | |
| 4 | Cherry | 13 | 14 | 9 | $105.30 | |
| 5 | Apple | 14 | 15 | 10 | $75.00 | |
| 6 | Pear | 9 | 8 | 8 | $76.80 | |
| 7 | Apple | 8 | 9 | 6 | $45.00 | |
| 8 | | | | | | |
| 9 | Tree | Height | Age | Yield | Profit | Height |
| 10 | Apple | >10 | | | | <16 |
| 11 | Pear | | | | | |

DSUM(Database,"Profit",A9:A10) equals $225.00, the total Profit from Apple trees.

DSUM(Database,"Profit",A9:F10) equals $75.00, the total Profit from Apple trees with a Height between 10 and 16.

DPRODUCT(Database,"Yield",A9:F10) equals 10, the product of the Yields from Apple trees with a Height between 10 and 16.

DAVERAGE(Database,3,Database) equals 13, the average Age of all trees in the database.

DAVERAGE(Database,"Yield",A9:A11) equals 9.6, the average Yield for Apple and Pear trees.

DSTDEV(Database,"Yield",A9:A11) equals 2.97, the estimated standard deviation in the Yield of Apple and Pear trees if the data in the database is only a sample of the total orchard population.

DSTDEVP(Database,"Yield",A9:A11) equals 2.65, the true standard deviation in the Yield of Apple and Pear trees if the data in the database is the entire population.

DVAR(Database,"Yield",A9:A11) equals 8.8, the estimated variance in the Yield of Apple and Pear trees if the data in the database is only a sample of the total orchard population.

DVARP(Database,"Yield",A9:A11) equals 7.04, the true variance in the Yield of Apple and Pear trees if the data in the database is the entire orchard population.

DGET(Database,"Yield",Criteria) returns the #NUM! error value because more than one record meets the criteria.

### Related Functions

AVERAGE, COUNT, COUNTA, MAX, MIN, PRODUCT, STDEV, STDEVP, SUM, VAR, and VARP perform the same operations as the corresponding database functions. However, they operate on their lists of arguments instead of on selected database entries.

## DGET(database,field,criteria)

Extracts single values from a database. Use DGET to extract a single record that matches the criteria from a database. If no record matches the criteria, DGET returns the #VALUE! error value. If more than one record matches the criteria, DGET returns the #NUM! error value. For more information on Microsoft Excel database functions, see "*Dfunction*."



*Macro Sheets Only*

## DIALOG.BOX(dialog_ref)

Displays the dialog box described in the range *dialog_ref* on your macro sheet or worksheet.

***Dialog_ref*** is a reference to a dialog-box definition table on a worksheet or macro sheet.

- The dialog-box definition table is described in more detail in Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.
- If the OK button in the dialog box is chosen, DIALOG.BOX enters values in fields as specified in the *dialog_ref* area and returns the item number of the button chosen.
- If the Cancel button in the dialog box is chosen, DIALOG.BOX returns FALSE.
- If *dialog_ref* is invalid, DIALOG.BOX returns the #VALUE! error value, and when the macro runs, a message is displayed that tells you which cell in *dialog_ref* contains the error.

The dialog-box definition table must be at least seven columns wide and two rows high. These are the definitions of each column in a dialog-box definition table:

| Column type | Column number |
|---|---|
| Item number | 1 |
| Horizontal position | 2 |
| Vertical position | 3 |
| Item width | 4 |
| Item height | 5 |
| Text | 6 |
| Initial value or result | 7 |

The first row of *dialog_ref* defines the size, position, type, and name of the dialog box. Dialog-box items are numbered starting with 1 for the second row in *dialog_ref*. For example, if the OK button item is in the 12th row of the dialog-box definition table, choosing OK would return 11, as the rows are numbered starting with the second row. These are the numbers for the items you can display in a dialog box:

| Dialog-box item | Item number |
|---|---|
| Default OK button | 1 |
| Cancel button | 2 |
| OK button | 3 |
| Default Cancel button | 4 |
| Static text | 5 |
| Text edit box | 6 |
| Integer edit box | 7 |
| Number edit box | 8 |
| Formula edit box | 9 |
| Reference edit box | 10 |
| Option button group | 11 |
| Option button | 12 |
| Check box | 13 |
| Group box | 14 |
| List box | 15 |
| Linked list box | 16 |
| Icons | 17 |
| Linked file list box (in Microsoft Excel for Windows or OS/2 only) | 18 |
| Linked drive and directory box (in Microsoft Excel for Windows or OS/2 only) | 19 |
| Directory text box | 20 |
| Drop-down list box | 21 |
| Drop-down combination edit/list box | 22 |

### Notes

- Microsoft Excel 3.0 supports dynamic user-defined dialog boxes. The Formula Define Name dialog box in Microsoft Excel is an example of a dynamic dialog box. You can type a name and a reference, select the Add button, and the name is added immediately to the Names In Sheet box. You can use dynamic user-defined dialog boxes to display immediately the results of choices made in the dialog box.
- You can use DIALOG.BOX(FALSE) to clear a dynamic dialog box from the screen if the macro has regained control from a trigger rather than from an OK button.

### Example

The following macro formula displays the dialog box described in the range InputDialog:

```
DIALOG.BOX(InputDialog)
```

Macro Sheets Only

## DIRECTORY(*path_text*)

Sets the current drive and directory or folder to the specified path and returns the name of the new directory or folder as text. Use DIRECTORY to get the name of the current directory or folder for use with the OPEN and SAVE.AS functions or to specify a directory or folder from which to return a list of files with the FILES function.

***Path_text*** is the drive and directory or folder you want to change to, enclosed in quotation marks.

- If *path_text* is not specified, DIRECTORY returns the name of the current directory or folder as text.
- If *path_text* does not include a drive specifier, the current drive is assumed.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula sets the directory to \EXCEL\MODELS on the current drive and returns the value "*drive*:\EXCEL\MODELS":

```
DIRECTORY("\EXCEL\MODELS")
```

The following macro formula sets the current drive to E and sets the directory to \EXCEL\MODELS on E. It returns the value "E:\EXCEL\MODELS":

```
DIRECTORY("E:\EXCEL\MODELS")
```

In Microsoft Excel for the Macintosh, the following macro formula sets the folder to Hard Disk:Apps:Excel:Financials and returns the value "Hard Disk:Apps:Excel:Financials":

```
DIRECTORY("Hard Disk:Apps:Excel:Financials")
```

### Related Function

| | |
|---|---|
| FILES | Returns a horizontal array of files in the current directory |

Macro Sheets Only

## DISABLE.INPUT(*logical*)

Blocks all input from the keyboard and mouse to Microsoft Excel (except input to displayed dialog boxes). Use DISABLE.INPUT to prevent input from the user (such as pressing ESC) or from other applications.

***Logical*** is a logical value specifying whether input is currently disabled. If *logical* is TRUE, input is disabled; if FALSE, input is reenabled.

Disabling input can be useful if you are using Dynamic Data Exchange (DDE) to communicate with Microsoft Excel from another application. For information on Dynamic Data Exchange, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*. For information on AppleEvents™, see Appendix D, "Using Microsoft Excel with System Software Version 7.0," in the *Microsoft Excel User's Guide* for the Macintosh.

### Important

Be sure to end any macro that uses DISABLE.INPUT(TRUE) with a DISABLE.INPUT(FALSE) function. If you do not include DISABLE.INPUT(FALSE) to allow non-dialog-box input, you will not be able to take any actions on your computer after the macro has finished.

### Related Function

| | |
|---|---|
| WORKSPACE | Changes workspace settings |



Macro Sheets Only

## DISPLAY(*formulas,gridlines,headings,zeros, color_num,reserved,outline,page_breaks,object_num*)

## DISPLAY?(*formulas,gridlines,headings,zeros, color_num,reserved,outline,page_breaks,object_num*)

**Form 1: For controlling screen display**

Equivalent to the Options Display command. Controls whether the screen displays formulas, gridlines, row and column headings, and other screen attributes. There are two forms of this function. Use the other form to control the display of the Info window. For detailed information on the Options Display command, see online Help.

Arguments for this form correspond to options and check boxes in the Options Display dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Formulas*** corresponds to the Formulas check box. The default is FALSE on worksheets and TRUE on macro sheets.

***Gridlines*** corresponds to the Gridlines check box. The default is TRUE.

***Headings*** corresponds to the Row & Column Headings check box. The default is TRUE.

***Zeros*** corresponds to the Zero Values check box. The default is TRUE.

***Color_num*** is a number from 0 to 16 corresponding to the gridline and heading colors in the Options Display dialog box; 0 corresponds to automatic color and is the default value.

***Reserved*** is reserved for certain international versions of Microsoft Excel.

***Outline*** corresponds to the Outline Symbols check box. The default is TRUE.

***Page_breaks*** corresponds to the Automatic Page Breaks check box. The default is FALSE.

***Object_num*** is a number from 1 to 3 corresponding to the display options in the Object box.

| *Object_num* | Corresponds to |
|---|---|
| 1 or omitted | Show All |
| 2 | Show Placeholders |
| 3 | Hide |

### Example

The following macro formula turns off gridlines and row and column headings, and leaves all other settings at the default values:

```
DISPLAY(,FALSE,FALSE)
```

### Related Function

| | |
|---|---|
| WORKSPACE | Changes workspace settings |

Macro Sheets Only

## DISPLAY(*cell,formula,value,format,protection, names,precedents,dependents,note*)

**Form 2: For controlling Info window display**

Equivalent to the commands on the Info menu. Controls which commands on the Info window are in effect. There are two forms of this function. Use the other form to control screen display. The Info window must be active to use this form of DISPLAY. For detailed information on the Info menu commands, see online Help.

Arguments in this form correspond to commands on the Info menu with the same names. For example,

*cell* corresponds to the Info Cell command. All the arguments except *precedents* and *dependents* are logical values. For these arguments:

- If the argument is TRUE, Microsoft Excel displays the corresponding Info item.
- If the argument is FALSE, Microsoft Excel does not display the corresponding Info item.
- If the argument is omitted, the status of the item is unchanged.

***Precedents*** is a number from 1 to 3 that specifies which precedents to list, according to the following table.

***Dependents*** is a number from 1 to 3 that specifies which dependents to list, according to the following table.

| *Precedents* or *dependents* | List |
|---|---|
| 0 | None |
| 1 | Direct only |
| 2 | All levels |

### Related Function

| | |
|---|---|
| SHOW.INFO | Controls the display of the Info window |

## DMAX(*database,field,criteria*)

Returns the largest number in the *field* column of records in the *database* which match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

## DMIN(*database,field,criteria*)

Returns the smallest number in the *field* column of records in the *database* which match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

*Macro Sheets Only*

## DOCUMENTS(*type_num*)

Returns, as a horizontal array of quoted text values, the names of all open documents in alphabetical order. Use DOCUMENTS to get the names of all open documents to use in other functions that manipulate open documents.

***Type_num*** is a number specifying whether to include add-in documents in the array of documents, according to the following table:

| *Type_num* | Returns |
|---|---|
| 1 or omitted | Names of all open documents except add-in documents |
| 2 | Names of add-in documents only |
| 3 | Names of all open documents |

### Tips

- Use the INDEX function to select individual document names from the array to use in other functions that take document names as arguments.
- Use COLUMNS to count the number of entries in the array.
- Use TRANSPOSE to change a horizontal array to a vertical one.

### Examples

In Microsoft Excel for Windows or OS/2, if your workspace contains windows named BUDGET.XLS, Chart1, ACTUAL.XLS:1, and ACTUAL.XLS:2, then:

DOCUMENTS(1) equals the three-cell array {"ACTUAL.XLS","BUDGET.XLS","Chart1"}

In Microsoft Excel for the Macintosh, if your workspace contains windows named Budget, Chart1, Actuals, and Actuals:2, then:

DOCUMENTS(1) equals the three-cell array {"Actuals","Budget","Chart1"}

The following macro hides all displayed documents:



| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | HideAllDocuments | |
| 4 | | =SET.NAME("Docs",DOCUMENTS()) | Names document array |
| 5 | | =FOR("Counter",1,COLUMNS(Docs)) | Begins FOR loop |
| 6 | Name | =INDEX(Docs,Counter) | Gets name of document |
| 7 | | =ACTIVATE(Name) | Activates the document |
| 8 | | =HIDE() | Hides the document |
| 9 | | =NEXT() | Ends the loop |
| 10 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| FILES | Returns a horizontal array of files in the current directory |
| GET.DOCUMENT | Returns information about a document |
| GET.WINDOW | Returns information about a window |
| WINDOWS | Returns a list of the open windows |

## DOLLAR(*number,decimals*)

Converts a number to text using currency format, with the decimals rounded to the specified place. The format used is $#,##0.00_);($#,##0.00).

***Number*** is a number, a reference to a cell containing a number, or a formula that evaluates to a number.

***Decimals*** is the number of digits to the right of the decimal point. If *decimals* is negative, *number* is rounded to the left of the decimal point. If you omit *decimals*, it is assumed to be 2.

### Note

The major difference between formatting a cell containing a number with the Format Number command and formatting a number directly with the DOLLAR function is that DOLLAR converts its result to text. A number formatted with the Format Number command is still a number.

You can continue to use numbers formatted with DOLLAR in formulas, because Microsoft Excel converts numbers entered as text values to numbers when it calculates.

### Examples

DOLLAR(1234.567,2) equals "$1,234.57"

DOLLAR(1234.567,–2) equals "$1,200"

DOLLAR(–1234.567,–2) equals "($1,200)"

DOLLAR(–0.123,4) equals "($0.1230)"

DOLLAR(99.888) equals "$99.89"

### Related Functions

| | |
|---|---|
| FIXED | Rounds and formats a number and converts it to text |
| TEXT | Formats a number and converts it to text |
| VALUE | Converts a text argument to a number |

## DPRODUCT(*database,field,criteria*)

Multiplies the values in the *field* column of records in the *database* that match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

## DSTDEV(*database,field,criteria*)

Estimates the standard deviation of a population based on a sample, using the numbers in the *field* column of records in the *database* that match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

## DSTDEVP(database,field,criteria)

Calculates the standard deviation of a population based on the entire population, using the numbers in the *field* column of records in the *database* that match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

## DSUM(database,field,criteria)

Adds the numbers in the *field* column of records in the *database* that match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

Macro Sheets Only

## DUPLICATE()

Duplicates the selected object. If an object is not selected, returns the #VALUE! error value and interrupts the macro.

### Related Functions

| | |
|---|---|
| COPY | Copies information to the Clipboard |
| PASTE | Pastes copied or cut data |

## DVAR(database,field,criteria)

Estimates the variance of a population based on a sample, using the numbers in the *field* column of records in the *database* that match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

## DVARP(database,field,criteria)

Calculates the variance of a population based on the entire population, using the numbers in the *field* column of records in the *database* that match the *criteria*. For more information on Microsoft Excel database functions, see "D*function*."

Macro Sheets Only

## ECHO(logical)

Controls screen updating while a macro is running. If a large macro uses many commands that update the screen, use ECHO to make the macro run faster.

**Logical** is a logical value specifying whether screen updating is on or off.

- If *logical* is TRUE, Microsoft Excel turns on screen updating.
- If *logical* is FALSE, Microsoft Excel turns off screen updating.
- If *logical* is omitted, Microsoft Excel changes the current screen update condition.

### Note

Screen updating is always turned back on when a macro ends.

### Example

See the example for PARSE.

Macro Sheets Only

## EDIT.COLOR(color_num,red_value,green_value,blue_value)

## EDIT.COLOR?(color_num,red_value,green_value,blue_value)

Equivalent to choosing the Edit button in the Options Color Palette dialog box. Defines the color for one of the 16 color palette boxes. Use EDIT.COLOR if you want to use a color that is not currently on the palette and if your system hardware has more than 16 colors available. After you set the color for the color box, any items previously formatted with that color are displayed in the new color. For detailed information on the Options Color Palette command, see online Help.

**Color_num** is a number from 1 to 16 specifying one of the 16 color palette boxes for which you want to set the color.

**Red value, green value,** and **blue value** are numbers that specify how much red, green, and blue are in each color.

- In Microsoft Excel for Windows or OS/2, *red_value*, *green_value*, and *blue_value* are numbers from 0 to 255.
- In Microsoft Excel for the Macintosh, *red_value*, *green_value*, and *blue_value* are also numbers from 0 to 255. However, the color editing dialog box displays numbers from 0 to 65,535. Microsoft Excel automatically converts the numbers between the two ranges. This allows you to display similar colors in all operating environments without modifying your macros.
- If *red_value*, *green_value*, and *blue_value* are all set to 255, the resulting color is white. If they are all set to zero, the resulting color is black.
- If *red_value*, *green_value*, or *blue_value* is omitted, Microsoft Excel assumes it to be the appropriate value for that *color_num*.

### Notes

- The number of unique colors that you can choose from and the number of colors that can be displayed on the screen at the same time are determined by your system hardware.
- EDIT.COLOR does not use hue, saturation, or luminance values. If you are using the macro recorder and set the color of a color palette box using hue, saturation, and luminance, Microsoft Excel records the corresponding red, green, and blue values instead.
- The dialog-box form of this function, EDIT.COLOR?(*color_num*), displays your system's color editing dialog box. The default *red_value*, *green_value*, and *blue_value* are determined by the current settings for the *color_num* you specify. *Color_num* is a required argument.

### Example

The following macro formula sets the first color box on the color palette to black:

```
EDIT.COLOR(1,0,0,0)
```

### Related Function

| | |
|---|---|
| COLOR.PALETTE | Copies a color palette from one document to another |

Macro Sheets Only

## EDIT.DELETE(shift_num)

## EDIT.DELETE?(shift_num)

Equivalent to the Edit Delete command. Removes the selected cells from the worksheet and shifts other cells to close up the space. For detailed information on the Edit Delete command, see online Help.

**Shift_num** is a number from 1 to 4 specifying whether to shift cells left or up after deleting the current selection or else to delete the entire row or column.

| Shift_num | Result |
|---|---|
| 1 | Shifts cells left |
| 2 | Shifts cells up |
| 3 | Deletes entire row |
| 4 | Deletes entire column |

- If *shift_num* is omitted and if one cell or a horizontal range is selected, EDIT.DELETE shifts cells up.
- If *shift_num* is omitted and a vertical range is selected, EDIT.DELETE shifts cells left.

### Related Function

| | |
|---|---|
| CLEAR | Removes data, formats, or both from selected cells |

Macro Sheets Only

## EDIT.REPEAT()

Equivalent to the Edit Repeat command. Repeats certain actions and commands. EDIT.REPEAT is available in the same situations as the Edit Repeat command. For detailed information on the Edit Repeat command, see online Help.

Macro Sheets Only

## EDIT.SERIES(*series_num,name_ref,x_ref,y_ref, z_ref,plot_order*)

## EDIT.SERIES?(*series_num,name_ref,x_ref,y_ref, z_ref,plot_order*)

Equivalent to the Chart Edit Series command. Creates or changes chart series by adding a new SERIES formula or modifying an existing SERIES formula in a chart. Use EDIT.SERIES when you want to change the data that is plotted in a chart.

***Series_num*** is the number of the series you want to change. If *series_num* is 0 or omitted, Microsoft Excel creates a new data series.

***Name_ref*** is the name of the data series. It can be an external reference to a single cell, a name defined as a single cell, or a name defined as a sequence of characters. *Name_ref* can also be text enclosed in quotation marks (for example, "Projected Sales").

***X_ref*** is an external reference to the name of the worksheet and the cells that contain one of the following sets of data:

- Category labels for all charts except xy (scatter) charts
- X-coordinate data for xy (scatter) charts

***Y_ref*** is an external reference to the name of the worksheet and the cells that contain values (or y-coordinate data in xy (scatter) charts) for all 2-D charts. *Y_ref* is required in 2-D charts but does not apply to 3-D charts.

***Z_ref*** is an external reference to the name of the worksheet and the cells that contain values for all 3-D charts. *Z_ref* is required in 3-D charts but does not apply to 2-D charts.

***Plot_order*** is a number specifying whether the data series is plotted first, second, and so on, in the chart.

- No two series can have the same *plot_order*.
- If you assign a *plot_order* to a series, Microsoft Excel plots that series in the order you specify, and the series that previously had that plot order (and any series following it) has its plot order increased by one.
- If you add a series to a chart with an overlay, the number of series in the main chart does not change, so if the series is added to the main chart, then the series that was plotted last in the main chart will be plotted first in the overlay chart. To change which series is plotted first in the overlay chart, use the Format Overlay command or the FORMAT.OVERLAY function.
- If you omit *plot_order* when you add a new series, then Microsoft Excel plots that series last and assigns it the correct *plot_order* value.
- The maximum value for *plot_order* is 255.

### Note

*X_ref*, *y_ref*, and *z_ref* can be arrays or references to a multiple selection, although they cannot be names that refer to a multiple selection. If you specify a multiple selection for any of these arguments, make sure you include the necessary sets of parentheses so that Microsoft Excel does not treat the components of the references as separate arguments. See the second example below.

### Tip

To delete a data series, use the SELECT and FORMULA macro functions as shown in the following portion of a macro, where *n* is the series number:

| | A | B | C |
|---|---|---|---|
| 5 | | =SELECT("S*n*") | |
| 6 | | =FORMULA("") | |



### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula changes the second data series on the active chart so that its category labels are in cells A2:A5, its y values are in cells C2:C5, and its plot order is 3. The series name is February Sales, and the data for the chart comes from a worksheet named SALES.XLS.

```
EDIT.SERIES(2,"February Sales",[SALES.XLS]$A$2:$A$5,
SALES.XLS!$C$2:$C$5,,3)
```

The following macro formula defines the second data series as the previous example does, except that the y-values are in the multiple selection A8:A9, A13:A14, and A20:A21.

```
EDIT.SERIES(2,"February Sales",SALES.XLS!$A$2:$A$5,
(SALES.XLS!$A$8:$A$9,SALES.XLS!$A$13:$A$14,
SALES.XLS!$A$20:$A$21),,3)
```

In Microsoft Excel for the Macintosh, the following macro formula changes the second data series on the active chart so that its category labels are in cells A2:A5, its y-values are in cells C2:C5, and its plot order is 3. The series name is February Sales, and the data for the chart comes from a worksheet named Sales.

```
EDIT.SERIES(2,"February Sales",Sales!$A$2:$A$5,
Sales!$C$2:$C$5,,3)
```

The following macro formula defines the second data series as the previous example does, except that the y-values are in the multiple selection A8:A9, A13:A14, and A20:A21.

```
EDIT.SERIES(2,"February Sales",Sales!$A$2:$A$5,
(Sales!$A$8:$A$9,Sales!$A$13:$A$14,Sales!$A$20:$A$21),,3)
```

Macro Sheets Only

## EDITION.OPTIONS(*edition_type, edition_name,reference,option*)

## EDITION.OPTIONS?(*edition_type, edition_name,reference*)

Sets options in or performs actions on the specified publisher or subscriber. In Microsoft Excel for Windows or OS/2, EDITION.OPTIONS allows you to cancel a publisher or subscriber created in Microsoft Excel for the Macintosh.

***Edition_type*** is the number 1 or 2 specifying the type of edition.

| *Edition_type* | Type of edition |
|---|---|
| 1 | Publisher |
| 2 | Subscriber |

***Edition_name*** is the name of the edition you want to change the edition options for or perform actions on. If *edition_name* is omitted, *reference* is required.

***Reference*** specifies the range (given in R1C1 format) occupied by the publisher or subscriber. *Reference* is required if you have more than one publisher or subscriber of *edition_name* on the active document. Use *reference* to specify the location of the publisher or subscriber for which you want to set options. If *edition_type* is 1 and the publisher is an embedded chart, or if *edition_type* is 2 and the subscriber is a picture, *reference* is the object identifier as displayed in the reference area. If *reference* is omitted, *edition_name* is required.

***Option*** is a number from 1 to 5 specifying the edition option you want to set or action you want to take, according to the following two tables. *Options* 2 to 5 are only available if you are using Microsoft Excel for the Macintosh with system software version 7.0 or later.

If a publisher is specified, then *option* applies as follows:

| *Option* | Action |
|---|---|
| 1 | Cancels the publisher |
| 2 | Sends the edition now |
| 3 | Selects the range or object published to the specified edition |
| 4 | Automatically updates the edition when the file is saved |
| 5 | Updates the edition on request only |

If a subscriber is specified, then *option* applies as follows:

| Option | Action |
|---|---|
| 1 | Cancels the subscriber |
| 2 | Gets the latest edition |
| 3 | Opens publisher document |
| 4 | Automatically updates when new data is available |
| 5 | Update on request only |

### Examples

The following macro formula opens the document (and application) that published the edition named Monthly Totals:

```
EDITION.OPTIONS(2,"Monthly Totals",,3)
```

The following macro checks the most recent edition date for a subscriber and, if that date is longer than three days ago, updates the edition:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | UpdateSubscriber | |
| 4 | Date | =GET.LINK.INFO("Subscriber 1",1,6) | Gets last update date |
| 5 | Date2 | =DATEVALUE(DATE) | Converts it to a value |
| 6 | | =IF(Date2<NOW()-2,,HALT()) | If more than 2 days ago... |
| 7 | | =EDITION.OPTIONS(2,"Subscriber 1",,2) | Updates the edition |
| 8 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| CREATE.PUBLISHER | Creates an edition of the selection |
| GET.LINK.INFO | Returns information about a link |
| SUBSCRIBE.TO | Inserts a published file into a document |

Macro Sheets Only

## ELSE()

Used with IF, ELSE.IF, and END.IF to control which functions are carried out in a macro. ELSE signals the beginning of a group of formulas in a macro sheet that will be carried out if the results of all preceding ELSE.IF statements and the preceding IF statement are FALSE. Use ELSE with IF, ELSE.IF, and END.IF when you want to perform multiple actions based on a condition. This method is preferable to using GOTO because it makes your macros more structured.

### Note

ELSE must be entered in a cell by itself. In other words, the cell can contain only "=ELSE()".

For more information on ELSE, ELSE.IF, END.IF, and IF, and for examples of these functions, see form 2 of the IF function later in this book.

### Related Functions

| | |
|---|---|
| ELSE.IF | Specifies a logical test to perform if the preceding IF or ELSE.IF function returns FALSE |
| END.IF | Ends a block of functions associated with the preceding IF or ELSE.IF function |
| IF | Specifies a logical test to perform and precedes the block of functions that will be carried out if the logical test is TRUE |

Macro Sheets Only

## ELSE.IF(logical_test)

Used with IF, ELSE, and END.IF to control which functions are carried out in a macro. ELSE.IF signals the beginning of a group of formulas in a macro sheet that will be carried out if the preceding IF or ELSE.IF function returns FALSE and if *logical_test* is TRUE. Use ELSE.IF with IF, ELSE, and END.IF when you want to perform multiple actions based on a condition. This method is preferable to using GOTO because it makes your macros more structured.

***Logical_test*** is a logical value that ELSE.IF uses to determine what functions to carry out next—that is, where to branch.

- If *logical_test* is TRUE, Microsoft Excel carries out the functions between the ELSE.IF function and the next ELSE.IF, ELSE, or END.IF function.
- If *logical_test* is FALSE, Microsoft Excel immediately branches to the next ELSE.IF, ELSE, or END.IF function.

### Notes

- ELSE.IF must be entered in a cell by itself.
- *Logical_test* will always be evaluated, even if the ELSE.IF section is not reached (due to a previous IF or ELSE.IF *logical_test* evaluating to TRUE). For this reason, you should not use formulas that carry out actions for *logical_test*. If you need to base the ELSE.IF condition on the return value of a formula that carries out an action, use the form "ELSE, IF(*logical_test*), and END.IF" in place of "ELSE.IF(*logical_test*)".

For more information on ELSE, ELSE.IF, END.IF, and IF, and for examples of these functions, see form 2 of the IF function later in this book.

### Related Functions

| | |
|---|---|
| CHOOSE | Chooses a value from a list of values or runs a macro from a list of macros |
| ELSE | Specifies what action to take if the preceding IF or ELSE.IF function returns FALSE |
| END.IF | Ends a block of functions associated with the preceding IF or ELSE.IF function |
| IF | Specifies a logical test to perform and precedes the block of functions that will be carried out if the logical test is TRUE |

## EMBED(object_type)

Displayed in the formula bar when an embedded object is selected. EMBED cannot be entered on a worksheet or used in a macro.

***Object_type*** is the name of the application and document type that created the embedded object. For example, the *object_types* used when Microsoft Excel documents are embedded in other applications are "ExcelWorksheet" and "ExcelChart".

### Note

If you delete the EMBED formula, the embedded object remains on the document as a graphic, and the link to the creating application is deleted. Double-clicking the object no longer starts the creating application.



Macro Sheets Only

## ENABLE.COMMAND(bar_num,menu,command,enable)

Enables or disables a custom command or menu. Disabled commands appear dimmed and can't be chosen. Use ENABLE.COMMAND to control which commands the user can choose in a menu bar.

***Bar_num*** is the menu bar in which a command resides. *Bar_num* can be the number of a built-in menu bar or the number returned by a previously run ADD.BAR function. See ADD.COMMAND for a list of the built-in bar numbers.

***Menu*** is the menu on which the command resides. *Menu* can be either the name of a menu as text or the number of a menu. Menus are numbered starting with 1 from the left of the screen.

***Command*** is the command you want to enable or disable. *Command* can be either the name of the command as text or the number of the command. The top command on a menu is command 1. If *command* is 0, ENABLE.COMMAND enables or disables the entire menu.

***Enable*** is a logical value specifying whether the command should be enabled or disabled. If *enable* is TRUE, Microsoft Excel enables the command; if FALSE, it disables the command.

### Note

You cannot disable built-in commands. If the specified command is a built-in command or does not exist, ENABLE.COMMAND returns the #VALUE! error value and interrupts the macro.

For information on custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Examples

The following formula disables a custom command that had been added previously to the Formula menu on the worksheet and macro sheet menu bar (full menus):

ENABLE.COMMAND(1,"Formula","Audit...",FALSE)

### Related Functions

| | |
|---|---|
| ADD.COMMAND | Adds a command to a menu |
| CHECK.COMMAND | Adds or deletes a check mark from a command |
| DELETE.COMMAND | Deletes a command from a menu |
| RENAME.COMMAND | Renames a command |

Macro Sheets Only

## END.IF()

Ends a block of functions associated with the preceding IF function. You must include one and only one END.IF function for each macro-sheets-only form (form 2) of the IF function in a macro. Form 1 of the IF function, which can be used on both worksheets and macro sheets, does not require an END.IF function. Use END.IF with IF, ELSE, and ELSE.IF when you want to perform multiple actions based on a condition. This method is preferable to using GOTO because it makes your macros more structured.

### Notes

- If you accidentally omit an END.IF function, your macro will end with an error at the cell containing the first IF function that does not have a corresponding END.IF function.
- END.IF must be entered in a cell by itself.

For more information on ELSE, ELSE.IF, END.IF, and IF, and for examples of these functions, see form 2 of the IF function later in this book.

### Related Functions

| | |
|---|---|
| ELSE | Specifies what action to take if the preceding IF or ELSE.IF function returns FALSE |
| ELSE.IF | Specifies a logical test to perform if the preceding IF or ELSE.IF function returns FALSE |
| IF | Specifies a logical test to perform and precedes the block of functions that will be carried out if the logical test is TRUE |

Macro Sheets Only

## ERROR(enable_logical,macro_ref)

Specifies what action to take if an error is encountered while a macro is running. Use ERROR to control whether Microsoft Excel error messages are displayed or to run your own macro when an error is encountered.

***Enable_logical*** is a logical value or number that turns error checking on or off.

- If *enable_logical* is FALSE or 0, all error checking is turned off. If error checking is turned off, and an error is encountered while a macro is running, Microsoft Excel ignores it and continues. Error checking is turned back on by a RETURN() statement in the same macro that turned error checking off, by an ERROR(TRUE) statement, or when the macro stops running.
- If *enable_logical* is TRUE or 1, you can either turn on normal error checking (by omitting the other argument) or specify a macro to run when an error is encountered by using the *macro_ref* argument. When normal error checking is active, a dialog box is displayed when an error is encountered. You can choose to halt the macro, start single-stepping through the macro, or continue running the macro normally.



- If *enable_logical* is 2 and *macro_ref* is omitted, error handling is normal except that if the user chooses the Cancel button in an alert message, FALSE is returned and the macro is not interrupted.
- If *enable_logical* is 2 and *macro_ref* is given, the macro goes to that *macro_ref* when an error is encountered. If the user chooses Cancel in an alert message, FALSE is returned and the macro is not interrupted.

***Macro_ref*** specifies a macro to run if *enable_logical* is TRUE, 1, or 2 and an error is encountered. It can be either the name of the macro or a cell reference. If *enable_logical* is FALSE or 0, *macro_ref* is ignored.

### Important

Both ERROR(FALSE) and ERROR(TRUE,*macro_ref*) keep Microsoft Excel from displaying any messages at all, including the message asking whether to save changes when you close an unsaved document. If you want alert messages but not error messages to be displayed, use ERROR(2,*macro_ref*).

### Examples

ERROR(FALSE) turns off error checking.

ERROR(TRUE,Recover) turns on error checking and runs the macro named Recover when an error is encountered.

In the following macro, ERROR(TRUE, ForceMenus) runs the macro ForceMenus if an error occurs in the current macro.

| | B | C |
|---|---|---|
| 1 | *Formulas* | *Comments* |
| 2 | | |
| 3 | AddEditCommand | |
| 4 | =ERROR(TRUE,ForceMenus) | Sets macro to run if error |
| 5 | =ADD.COMMAND(2,"Edit",EditCommand) | Tries to add a command |
| 6 | =RETURN() | |

Also see FREAD and the second example in ARGUMENT.

### Related Functions

| | |
|---|---|
| CANCEL.KEY | Specifies an action to take when a cancel key is pressed |
| LAST.ERROR | Returns the reference of the cell where the last error occurred |
| ON.KEY | Runs a macro when a specified key is pressed |

## EXACT(text1,text2)

Compares two text strings and returns TRUE if they are exactly the same, FALSE otherwise. EXACT is case sensitive but ignores formatting differences. Use EXACT to test text being entered onto a document.

***Text1*** is the first text string, enclosed in quotation marks.

***Text2*** is the second text string, enclosed in quotation marks.

### Examples

EXACT("word","word") equals TRUE

EXACT("Word","word") equals FALSE

EXACT("w ord","word") equals FALSE

To make sure that a user-entered value matches a value in a range, enter the following formula as an array in a cell. To enter a formula as an array in a single cell, press CTRL+SHIFT+ENTER (in Microsoft Excel for Windows or OS/2) or COMMAND+ENTER (in Microsoft Excel for the Macintosh). The name TestValue refers to a cell containing a user-entered value; the name CompareRange refers to a list of text values to be checked.

{=OR(EXACT(TestValue,CompareRange))}

### Related Functions

| | |
|---|---|
| LEN | Returns the length of a text string |
| SEARCH | Finds one text value within another |

Macro Sheets Only

## EXEC(program_text,window_num)

**Form 1: For Microsoft Excel for Windows or OS/2**

## EXEC(program_text,,background, preferred_size_only)

**Form 2: For Microsoft Excel for the Macintosh**

Starts a separate program. Use EXEC to start other programs with which you want to communicate. Use EXEC with Microsoft Excel's other DDE functions (INITIATE, EXECUTE, and SEND.KEYS) to create a channel to another program and to send keystrokes and commands to the program. (SEND.KEYS is available only in Microsoft Excel for Windows or OS/2.)

The first form of this function is for Microsoft Excel for Windows or OS/2. The second form is for Microsoft Excel for the Macintosh.

### Important

Microsoft Excel for the Macintosh requires system software version 7.0 or later for the last two arguments of this function.

***Program_text*** is the name, as a quoted text string, of any executable file or, in Microsoft Excel for Windows or OS/2, any data file that is associated with an executable file.

- Use pathnames when the file or program to be started is not in the current directory or folder.
- In Microsoft Excel for Windows or OS/2, *program_text* can include any arguments and switches that are accepted by the program to be started. Also, if *program_text* is the name of a file associated with a specific installed program, EXEC starts the program and loads the specified file.

***Window_num*** is a number from 1 to 3 that specifies how the window containing the program should appear. *Window_num* is only available for use with Microsoft Excel for Windows or OS/2. The *window_num* argument is allowed on the Macintosh, but it is ignored.

| *Window_num* | Window appears |
|---|---|
| 1 | Normal size |
| 2 or omitted | Minimized size |
| 3 | Maximized size |

***Background*** is a logical value that determines whether the program specified by *program_text* is opened as the active program or in the background, leaving Microsoft Excel as the active program. If *background* is TRUE, the program is started in the background; if FALSE or omitted, the program is started in the foreground. *Background* is only available for use with Microsoft Excel for the Macintosh and system software version 7.0 or later.

***Preferred_size_only*** is a logical value that determines the amount of memory allocated to the program. If *preferred_size_only* is TRUE, the program is opened with its preferred memory allocation; if FALSE or omitted, it opens with the available memory if greater than its minimum requirement. *Preferred_size_only* is only available for use with Microsoft Excel for the Macintosh and system software version 7.0 or later. For information about changing the preferred memory size, see your Macintosh documentation.

### Note

If the EXEC function is successful, it returns the task ID number of the started program. The task ID number is a unique number that identifies a program. Use the task ID number in other macro functions, such as INITIATE, to refer to the program. If the EXEC function is unsuccessful, it returns the #VALUE! error value.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula starts Microsoft Word and loads the document SALES.DOC:

```
EXEC("SALES.DOC")
```



The following macro formula starts the program SEARCH.EXE. Use pathnames when the file or program to be started is not in the current directory:

```
EXEC("C:\WINDOWS\SEARCH.EXE")
```

The following macro formula starts Microsoft® Word for Windows and loads the document SALES.DOC:

```
EXEC("WINWORD.EXE SALES.DOC")
```

The following portion of a macro opens the file MEMO1.DOC in Microsoft Word for Windows, opens a communications channel to the document, and sends commands and keystrokes to Word for Windows.

| | A | B | C |
|---|---|---|---|
| 5 | | =EXEC("C:\WINWORD.EXE","C:MEMO1.DOC") | Starts Word, opens MEMO1 |
| 6 | Num | =INITIATE("WINWORD","C:MEMO1.DOC") | Opens a channel to same |
| 7 | | =EXECUTE(Num,"%I-") | Chooses Format Character |
| 8 | | =SEND.KEYS("%b",FALSE) | Sends ALT+b to Word |

In Microsoft Excel for the Macintosh, the following macro formula, where Sales is a Microsoft® Word document, starts Word and loads the document Sales:

```
EXEC("Sales")
```

The following macro formula starts Word:

```
EXEC("Hard Disk Apps:Word")
```

### Related Functions

| | |
|---|---|
| APPACTIVATE | Activates an application |
| EXECUTE | Carries out specified commands in an external application |
| INITIATE | Opens a channel to another application |
| SEND.KEYS | Sends keystrokes to another application (Microsoft Excel for Windows or OS/2 only) |
| TERMINATE | Closes a channel to another application |

Macro Sheets Only

## EXECUTE(channel_num,execute_text)

Carries out commands in another program. Use with EXEC, INITIATE, and SEND.KEYS to run another program through Microsoft Excel. (SEND.KEYS is available only in Microsoft Excel for Windows or OS/2.)

### Important

Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

***Channel_num*** is a number returned by a previously run INITIATE function. *Channel_num* refers to a channel through which Microsoft Excel communicates with another program.

***Execute_text*** is a quoted text string representing commands you want to carry out in the program specified by *channel_num*. The form of *execute_text* depends on the program you are referring to. To include specific key sequences in *execute_text*, use the format described under *key_text* in the ON.KEY function.

For information on accessing other programs, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*.

If EXECUTE is not successful, it returns one of the following error values:

| Value returned | Situation |
|---|---|
| #VALUE! | *Channel_num* is not a valid channel number. |
| #N/A | The program you are accessing is busy. |
| #DIV/0! | The program you are accessing does not respond after a certain length of time and you have pressed ESC to cancel. |
| #REF! | The EXECUTE request is refused. |

### Note

Commands sent to another program with EXECUTE will not work when a dialog box is displayed in the program. In Microsoft Excel for Windows or OS/2, you can use SEND.KEYS to send commands that make selections in a dialog box.

### Example

The following formula sends the number 25 and a carriage return to the application identified by *channel_num* 14:

```
EXECUTE(14,"25~")
```

For another example of EXECUTE, see the EXEC function.

### Related Functions

| | |
|---|---|
| EXEC | Starts another application |
| INITIATE | Opens a channel to another application |
| POKE | Sends data to another application |
| REQUEST | Returns data from another application |
| SEND.KEYS | Sends keystrokes to another application (Microsoft Excel for Windows or OS/2 only) |
| TERMINATE | Closes a channel to another application |

## EXP(*number*)

Returns e raised to the power of *number*. The constant e equals 2.71828182845904, the base of the natural logarithm.

***Number*** is the exponent applied to the base e.

### Notes

- To calculate powers of other bases, use the exponentiation operator (^).
- EXP is the inverse of LN, the natural logarithm of *number*.

### Examples

EXP(1) equals 2.71828182828 (the approximate value of e)

EXP(2) equals $e^2$ equals 7.38905609

EXP(LN(3)) equals 3

### Related Functions

| | |
|---|---|
| LN | Returns the natural logarithm of a number |
| LOG | Returns the logarithm of a number to a specified base |

*Macro Sheets Only*

## EXTRACT(*unique*)
## EXTRACT?(*unique*)

Equivalent to the Data Extract command (full menus). Finds database records that match the criteria defined in the criteria range and copies them into a separate extract range. For detailed information on the Data Extract command, see online Help.

***Unique*** is a logical value corresponding to the Unique Records Only check box in the Data Extract dialog box.

- If *unique* is TRUE, Microsoft Excel turns on the check box and excludes duplicate records from the extract list.
- If *unique* is FALSE or omitted, Microsoft Excel turns off the check box and extracts all records matching the criteria.

### Example

The following macro extracts unique records into two extract ranges based on two sets of criteria. The quoted names (Criteria1, ExtractRange1, and so on) indicate names on the external worksheet.



| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **ExtractMultipleRanges** | |
| 4 | | =DEFINE.NAME("Criteria","=Criteria1") | Defines first criteria |
| 5 | | =FORMULA.GOTO("ExtractRange1") | Selects first extract range |
| 6 | | =EXTRACT(TRUE) | Extracts unique records |
| 7 | | =DEFINE.NAME("Criteria","=Criteria2") | Defines second criteria |
| 8 | | =FORMULA.GOTO("ExtractRange2") | Selects second extract range |
| 9 | | =EXTRACT(TRUE) | |
| 10 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| DATA.FIND | Finds records in a database |
| SET.CRITERIA | Defines the name Criteria for the selected range on the active sheet |
| SET.DATABASE | Defines the name Database for the selected range on the active sheet |
| SET.EXTRACT | Defines the name Extract for the selected range on the active sheet |

## FACT(*number*)

Returns the factorial of a number. The factorial of a number is equal to 1*2*3*...* *number*.

***Number*** is the nonnegative number you want the factorial of. If *number* is not an integer, it is truncated.

### Examples

FACT(1) equals 1

FACT(1.9) equals FACT(1) equals 1

FACT(0) equals 1

FACT(–1) equals #NUM!

FACT(5) equals 1*2*3*4*5 equals 120

### Related Functions

| | |
|---|---|
| DPRODUCT | Multiplies the values in a particular field of records in a database that match the criteria |
| PRODUCT | Multiplies its arguments |

## FALSE()

Returns the logical value FALSE.

### Note

You can also type the word FALSE directly into the worksheet or formula, and Microsoft Excel interprets it as the logical value FALSE.

*Macro Sheets Only*

## FCLOSE(*file_num*)

Closes the specified file.

***File_num*** is the number of the file you want to close. *File_num* is returned by the FOPEN function that originally opened the file. If *file_num* is not a valid file number, FCLOSE returns the #VALUE! error value.

### Example

See the examples for FOPEN and RESULT.

### Related Functions

| | |
|---|---|
| CLOSE | Closes the active window |
| FILE.CLOSE | Closes the active file |
| FOPEN | Opens the specified file |

Macro Sheets Only

## FILE.CLOSE(*save_logical*)

Equivalent to the File Close command. Closes the active document. Use FILE.CLOSE instead of CLOSE when you want to close an entire document instead of just one of its windows.

***Save_logical*** is a logical value specifying whether to save the file before closing it.

| *Save_logical* | Result |
|---|---|
| TRUE | Saves the file |
| FALSE | Does not save the file |
| Omitted | If you've made changes to the file, displays a dialog box asking if you want to save the file |

### Related Functions

| | |
|---|---|
| CLOSE | Closes the active window |
| CLOSE.ALL | Closes all unprotected windows |
| FCLOSE | Closes the specified file |

Macro Sheets Only

## FILE.DELETE(*file_text*)
## FILE.DELETE?(*file_text*)

Equivalent to the File Delete command (full menus). Deletes a file from the disk. Although you will normally delete files manually, you can, for example, use FILE.DELETE in a macro to delete temporary files created by the macro.

***File_text*** is the name of the file to delete.

### Notes

- If Microsoft Excel can't find *file_text*, it displays a message saying that it cannot delete the file. To avoid this, include the entire pathname in *file_text*. See the second and fifth examples below. You can also use FILES to generate an array of filenames and then check if the file you want to delete is in the array. For an example of how to see if an entry is in an array, see OR later in this book.
- If a file is open when you delete it, the file is removed from the disk but remains open in Microsoft Excel.
- In the dialog box form, FILE.DELETE?, you can use an asterisk (*) to represent any series of characters and a question mark (?) to represent any single character. See the third and sixth examples below.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula deletes a file called CHART1.XLC from the current directory:

```
FILE.DELETE("CHART1.XLC")
```

The following macro formula deletes a file called 89INFO.XLS kept in a series of subdirectories:

```
FILE.DELETE("C:\EXCEL\SALES\89INFO.XLS")
```

The following macro formula displays the File Delete dialog box listing all documents whose extensions begin with the letters "XL":

```
FILE.DELETE?("*.XL?")
```

In Microsoft Excel for the Macintosh, the following macro formula deletes a file called Chart1 from the current folder:

```
FILE.DELETE("Chart1")
```

The following macro formula deletes a file called 1989 Info kept in a series of nested folders:

```
FILE.DELETE("Hard Disk:Excel 3:Sales Worksheets:
1989 Info")
```

The following macro formula displays the File Delete dialog box listing all documents beginning with the word Clients:

```
FILE.DELETE?("Clients*")
```

### Related Functions

| | |
|---|---|
| FILE.CLOSE | Closes the active file |
| FILES | Returns a horizontal array of files in the specified directory |



Macro Sheets Only

## FILES(*directory_text*)

Returns a horizontal text array of the names of all files in the specified directory or folder. Use FILES to build a list of filenames upon which you want your macro to operate.

***Directory_text*** specifies which directories or folders to return filenames from.

- *Directory_text* accepts an asterisk (*) to represent a series of characters and a question mark (?) to represent a single character in filenames.
- If *directory_text* is not specified, FILES returns filenames from the current directory.

FILES can return up to 255 filenames in an array.

### Tips

- You can use the COLUMNS functions to count the number of entries in the returned array.
- You can use the TRANSPOSE function to change a horizontal array to a vertical one.

### Examples

The following macro formula returns the names of all files in the current directory or folder:

```
FILES()
```

The following macro formula returns the names of all files starting with the letter F in the current directory or folder:

```
FILES("F*")
```

When entered as an array formula in several cells, the following macro formula returns to those cells the filenames in the current directory. If the directory contains fewer files than can fit in the selected cells, the #N/A error value appears in the extra cells.

```
FILES()
```

In Microsoft Excel for Windows or OS/2, the following macro formula returns all files starting with "SALE" and ending with the .XLC extension in the \EXCEL\CHART subdirectory:

```
FILES("C:\EXCEL\CHARTS\SALE*.XLC")
```

In Microsoft Excel for the Macintosh, the following macro formula returns all files starting with "Sale" in the nested Chart folder:

```
FILES("Disk:Excel:Chart:Sale*")
```

The following macro opens all files matching *directory_text:*

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | OpenFiles | |
| 4 | DirText | =INPUT("Enter the files to open",2) | Text can include wildcards |
| 5 | | =SET.NAME("FileArray",FILES(DirText)) | Enter in horizontal array |
| 6 | | =IF(ISERROR(COLUMNS(FileArray))) | If empty, no columns |
| 7 | | =ALERT("No files match directory text",2) | |
| 8 | | =ELSE() | Otherwise, matching files |
| 9 | | =FOR("Index",1,COLUMNS(FileArray)) | Loops once for each |
| 10 | NextFile | =INDEX(FileArray,1,Index) | Retreives the next filename |
| 11 | | =OPEN(NextFile) | And opens the file |
| 12 | | =NEXT() | |
| 13 | | =END.IF() | |
| 14 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| FILE.DELETE? | Displays a dialog box listing files to delete |
| OPEN? | Displays a dialog box listing files to open |

*Macro Sheets Only*

## FILL.DOWN()
## FILL.LEFT()
## FILL.RIGHT()
## FILL.UP()

Equivalent to the Edit Fill Down, Edit Fill Left, Edit Fill Right, and Edit Fill Up commands, respectively. For detailed information on the Edit Fill commands, see online Help.

FILL.DOWN copies the contents and formats of the cells in the top row of a selection into the rest of the rows in the selection.

FILL.LEFT copies the contents and formats of the cells in the right column of a selection into the rest of the columns in the selection.

FILL.RIGHT copies the contents and formats of the cells in the left column of a selection into the rest of the columns in the selection.

FILL.UP copies the contents and formats of the cells in the bottom row of a selection into the rest of the rows in the selection.

### Notes

- If you have a multiple selection, each range in the selection is filled separately with the contents of the source range.
- To choose the Edit Fill Left and Edit Fill Up commands, press Shift while selecting the Edit menu.

### Related Functions

| | |
|---|---|
| COPY | Copies information to the Clipboard |
| FORMULA.FILL | Fills a range with a formula |

*Macro Sheets Only*

## FILL.WORKGROUP(type_num)
## FILL.WORKGROUP?(type_num)

Equivalent to the Edit Fill Workgroup command. Fills the contents of the active worksheet's selection to the same area on all other worksheets in the workgroup. Use FILL.WORKGROUP to fill a range of cells on all worksheets in a workgroup at once. For detailed information on the Edit Fill Workgroup command, see online Help.

***Type_num*** is a number from 1 to 3 that corresponds to the choices in the Edit Fill Workgroup dialog box.

| *Type_num* | Type of information filled |
|---|---|
| 1 | All |
| 2 | Formulas |
| 3 | Formats |

### Related Function

| | |
|---|---|
| WORKGROUP | Creates a workgroup from an array of worksheet names |

## FIND(find_text,within_text,start_at_num)

Finds one string of text within another string of text and returns the number of the character at which *find_text* first occurs. You can also use SEARCH to find one string of text within another, but unlike SEARCH, FIND is case sensitive and doesn't allow wildcard characters.

***Find_text*** is the text you want to find.

- If *find_text* is "" (empty text), FIND matches the first character in the search string (that is, the character numbered *start_at_num* or 1).
- *Find_text* cannot contain any wildcard characters.

***Within_text*** is the text containing the text you want to find.

***Start_at_num*** specifies the character at which to start the search. The first character in *within_text* is character number 1. If you omit *start_at_num*, it is assumed to be 1.



### Note

FIND returns the #VALUE! error value if

- *Find_text* does not appear in *within_text*.
- *Start_at_num* is not greater than zero.
- *Start_at_num* is greater than the length of *within_text*.

### Examples

FIND("M","Miriam McGovern") equals 1

FIND("m","Miriam McGovern") equals 6

FIND("M","Miriam McGovern",3) equals 8

Suppose you have a list of parts and serial numbers in a worksheet, and you want to extract the names of the parts, but not the serial numbers, from each cell. You can use the FIND function to find the # symbol, and the MID function to omit the serial number. In the following worksheet, starting from row 2, each cell in column B contains these formulas using relative A1-style references:

MID(A2,1,FIND(" #",A2,1)–1) in cell B2

MID(A3,1,FIND(" #",A3,1)–1) in cell B3

and so on for the other cells in column B.

| | A | B |
|---|---|---|
| 1 | With serial number | Without serial number |
| 2 | Ceramic Insulators #124-TD45-87 | Ceramic Insulators |
| 3 | Copper Coils #12-671-6772 | Copper Coils |
| 4 | Variable Resistors #116010 | Variable Resistors |
| 5 | Blank Circuit Boards #44XT-56889-3 | Blank Circuit Boards |
| 6 | AND Gate Arrays #8TRF995-4455-X1 | AND Gate Arrays |
| 7 | 7 Segment LED Displays #11W-81524-2 | 7 Segment LED Displays |
| 8 | 8 Input Toggle Switches #21-555-sdH44 | 8 Input Toggle Switches |

### Related Functions

| | |
|---|---|
| EXACT | Checks to see if two text values are identical |
| LEN | Returns the length of a text string |
| MID | Extracts characters based on a starting position and number of characters |
| SEARCH | Finds one text value within another |

## FIXED(number,decimals)

Rounds a number to the specified number of decimals, formats the number in decimal format using a period and commas, and returns the result as text.

***Number*** is the number you want to round and convert to text.

***Decimals*** is the number of digits to the right of the decimal point.

- Numbers in Microsoft Excel can never have more than 15 significant digits, but *decimals* can be as large as 127.
- If *decimals* is negative, *number* is rounded to the left of the decimal point.
- If you omit *decimals*, it is assumed to be 2.

### Note

The major difference between formatting a cell containing a number with the Format Number command and formatting a number directly with the FIXED function is that FIXED converts its result to text. A number formatted with the Format Number command is still a number.

### Examples

FIXED(1234.567,1) equals "1,234.6"

FIXED(1234.567,–1) equals "1,230"

FIXED(–1234.567,–1) equals "–1,230"

FIXED(44.332) equals "44.33"

### Related Functions

| | |
|---|---|
| DOLLAR | Formats a number and converts it to text |
| ROUND | Rounds a number to a specified number of digits |
| TEXT | Formats a number and converts it to text |
| VALUE | Converts a text argument to a number |

*Macro Sheets Only*

## FONT(name_text,size_num)
## FONT?(name_text,size_num)

Equivalent to the Options Font command in Microsoft Excel for the Macintosh version 1.5 and is included only for macro compatibility. Sets the font for the Normal style. Microsoft Excel now uses the FORMAT.FONT and DEFINE.STYLE functions. For more information, see FORMAT.FONT and DEFINE.STYLE.

***Name_text*** is the name of the font in the form of text, such as "Courier".

***Size_num*** is the point size of the font.

### Related Functions

| | |
|---|---|
| DEFINE.STYLE | Creates or changes a cell style |
| FORMAT.FONT | Applies a font to the current selection |

*Macro Sheets Only*

## FOPEN(file_text,access_num)

Opens a file with the type of permission specified. Unlike OPEN, FOPEN does not load the file into memory and display it; instead, FOPEN establishes a channel with the file so that you can exchange information with it. If the file is opened successfully, FOPEN returns a file ID number. If it can't open the file, FOPEN returns the #N/A error value. Use the file ID number with other file functions (such as FREAD, FWRITE, and FSIZE) when you want to get information from or send information to the file.

***File_text*** is the name of the file you want to open.

***Access_num*** is a number from 1 to 3 specifying what type of permission to allow to the file:

| *Access_num* | Type of permission |
|---|---|
| 1 or omitted | Can read and write to the file (read/write permission) |
| 2 | Can read the file, but can't write to the file (read-only permission) |
| 3 | Creates a new file with read/write permission |

- If the file doesn't exist and *access_num* is 3, FOPEN creates a new file.
- If the file does exist and *access_num* is 3, FOPEN replaces the contents of the file with any information you supply using the FWRITE or FWRITELN functions.
- If the file doesn't exist and *access_num* is 1 or 2, FOPEN returns the #N/A error value.

### Note

Use FCLOSE to close a file after you finish using it.

### Example

The following macro opens a file, writes the contents of the selected nonblank cells on the active worksheet to the file, and then closes the file.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | WriteSelection | Writes cells to text file |
| 4 | FileName | =INPUT("Name of file to write:",2) | Asks for a text filename |
| 5 | FileNumber | =FOPEN(FileName,3) | Creates read/write file |
| 6 | | =FOR.CELL("CurrentCell",,TRUE) | Loops for nonblank cells |
| 7 | | =FWRITE(FileNumber,CurrentCell) | Writes contents to the file |
| 8 | | =NEXT() | |
| 9 | | =FCLOSE(FileNumber) | |
| 10 | | =RETURN() | |

Notice that FCLOSE and FWRITE depend on the value that FOPEN returns in cell B5.



For more examples of FOPEN, see FPOS, FREAD, and RESULT.

### Related Functions

| | |
|---|---|
| FCLOSE | Closes a text file |
| FREAD | Reads characters from a text file |
| FWRITE | Writes characters to a text file |
| OPEN | Opens an existing file or workspace |

*Macro Sheets Only*

## FOR(counter_text,start_num,end_num,step_num)

Starts a FOR–NEXT loop. The instructions between FOR and NEXT are repeated until the loop counter reaches a specified value. Use FOR when you need to repeat instructions a specified number of times. Use FOR.CELL when you need to repeat instructions over a range of cells.

***Counter_text*** is the name of the loop counter in the form of text.

***Start_num*** is the value initially assigned to *counter_text*.

***End_num*** is the last value assigned to *counter_text*.

***Step_num*** is a value added to the loop counter after each iteration. If *step_num* is omitted, it is assumed to be 1.

### Notes

- Microsoft Excel follows these steps as it executes a FOR–NEXT loop:

| Step | Action |
|---|---|
| 1 | Sets *counter_text* to the value *start_num*. |
| 2 | If *counter_text* is greater than *end_num* (or less than *end_num* if *step_num* is negative), the loop ends, and the macro continues with the function after the NEXT function.<br>If *counter_text* is less than or equal to *end_num* (or greater than or equal to *end_num* if *step_num* is negative), the macro continues in the loop. |
| 3 | Carries out functions up to the following NEXT function. The NEXT function must be below the FOR function and in the same column. |
| 4 | Adds *step_num* to the loop counter. |
| 5 | Returns to the FOR function and proceeds as described in step 2. |

- You can interrupt a FOR–NEXT loop by using the BREAK function.
- For more information on looping in macros, see Chapter 19, "Designing and Writing a Command Macro," in the *Microsoft Excel User's Guide*.

### Example

The following macro uses a FOR–NEXT loop to unhide all hidden windows:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | UnhideAllWindows | |
| 4 | | =SET.NAME("WinArray",WINDOWS()) | Names of open windows |
| 5 | | =FOR("Counter",1,COLUMNS(WinArray)) | Begins a FOR–NEXT loop |
| 6 | WinName | =INDEX(WinArray,1,Counter) | Name of one window |
| 7 | Hidden | =GET.WINDOW(7,WinName) | Hidden sheets? |
| 8 | | =IF(Hidden,UNHIDE(WinName)) | Unhides hidden sheets |
| 9 | | =NEXT() | Continues the loop |
| 10 | | =RETURN() | |

For more examples of FOR, see FILES, FREAD, and PARSE.

### Related Functions

| | |
|---|---|
| BREAK | Interrupts a FOR–NEXT, WHILE–NEXT, or FOR.CELL–NEXT loop |
| FOR.CELL | Begins a FOR.CELL–NEXT loop |
| NEXT | Ends a FOR–NEXT, FOR.CELL–NEXT, or WHILE–NEXT loop |
| WHILE | Begins a WHILE–NEXT loop |

*Macro Sheets Only*

## FOR.CELL(*ref_name,area_ref,skip_blanks*)

Starts a FOR.CELL–NEXT loop. This function is similar to FOR, except that the instructions between FOR.CELL and NEXT are repeated over a range of cells, one cell at a time, and there is no loop counter.

***Ref_name*** is the name in the form of text that Microsoft Excel gives to the one cell in the range that is currently being operated on; *ref_name* refers to a new cell during each loop.

***Area_ref*** is the range of cells on which you want the FOR.CELL–NEXT loop to operate and can be a multiple selection. If *area_ref* is omitted, it is assumed to be the current selection.

***Skip_blanks*** is a logical value specifying whether Microsoft Excel skips blank cells as it operates on the cells in *area_ref*.

| *Skip_blanks* | Result |
|---|---|
| TRUE | Skips blank cells in *area_ref* |
| FALSE or omitted | Operates on all cells in *area_ref* |

### Note

FOR.CELL operates on each cell in a row from left to right one area at a time before moving to the next row in the selection.

### Example

The following macro replaces the contents of each nonblank cell in the current selection with its cube root. This macro assumes that the appropriate worksheet is active and that each cell in the current selection contains a number.

| | A | B | C |
|---|---|---|---|
| 1 | *Macros* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | CubeRoot | Calculates the cube root |
| 4 | | =FOR.CELL("CurrentCell",TRUE) | Loops for each nonblank cell |
| 5 | | =FORMULA(CurrentCell^(1/3),CurrentCell) | Cube root of CurrentCell |
| 6 | | =NEXT() | |
| 7 | | =RETURN() | |

For other examples of FOR.CELL, see FOPEN and FPOS.

### Related Functions

| | |
|---|---|
| BREAK | Interrupts a FOR–NEXT, WHILE–NEXT, or FOR.CELL–NEXT loop |
| FOR | Begins a FOR–NEXT loop |
| NEXT | Ends a FOR–NEXT, FOR.CELL–NEXT, or WHILE–NEXT loop |
| WHILE | Begins a WHILE–NEXT loop |

*Macro Sheets Only*

## FORMAT.FONT(*name_text,size_num,bold,italic,underline,strike,color,outline,shadow*)

## FORMAT.FONT?(*name_text,size_num,bold,italic,underline,strike,color,outline,shadow*)

**Form 1: Cells**

## FORMAT.FONT(*name_text,size_num,bold,italic,underline,strike,color,outline,shadow,object_id_text,start_num,char_num*),

## FORMAT.FONT?(*name_text,size_num,bold,italic,underline,strike,color,outline,shadow,object_id_text,start_num,char_num*)

**Form 2: Text boxes and buttons on worksheets and macro sheets**



## FORMAT.FONT(*color,backgd,apply,name_text,size_num,bold,italic,underline,strike,outline,shadow*)

## FORMAT.FONT?(*color,backgd.apply,name_text,size_num,bold,italic,underline,strike,outline,shadow*)

**Form 3: Chart items including unattached chart text**

Equivalent to the Format Font command. Applies a font to the selection. FORMAT.FONT has three forms. The first form is for cells, the second form is for text boxes; the third form is used with all chart items (axes, labels, text, and so on). For detailed information on the Format Font command, see online Help.

Arguments correspond to check boxes or options in the Format Font dialog box. Arguments that correspond to check boxes are logical values. If the argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box. If an argument is omitted, the format is not changed.

***Name_text*** is the name of the font as it appears in the Format Font dialog box. For example, Courier is a font name.

***Size_num*** is the font size, in points.

***Bold*** corresponds to the Bold check box.

***Italic*** corresponds to the Italic check box.

***Underline*** corresponds to the Underline check box.

***Strike*** corresponds to the Strikeout check box.

***Color*** is a number from 0 to 16 corresponding to the colors in the Format Font dialog box; 0 corresponds to automatic color.

***Outline*** corresponds to the Outline check box. Outline fonts are available in Microsoft Excel for the Macintosh. For macro compatibility, the argument is ignored by Microsoft Excel for Windows or OS/2.

***Shadow*** corresponds to the Shadow check box. Shadow fonts are available in Microsoft Excel for the Macintosh. For macro compatibility, the argument is ignored by Microsoft Excel for Windows or OS/2.

***Object_id_text*** identifies the text box you want to format (for example, "Text 1", "Text 2", and so on). You can also use the object number alone without the text identifier. If this argument is omitted, Microsoft Excel formats text in the currently selected text box.

***Start_num*** specifies the first character to be formatted. If *start_num* is omitted, it is assumed to be 1 (the first character in the text box).

***Char_num*** specifies how many characters to format. If *char_num* is omitted, Microsoft Excel formats all characters in the text box starting at *start_num*.

***Backgd*** is a number from 1 to 3 specifying which type of background to apply to text in a chart.

| *Backgd* | Type of background applied |
|---|---|
| 1 | Automatic |
| 2 | Transparent |
| 3 | White Out |

***Apply*** corresponds to the Apply To All check box. This argument applies to data labels only.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula formats the selected range of cells with the 10-point Courier bold italic font:

```
FORMAT.FONT("Courier",10,TRUE,TRUE,FALSE,FALSE,0)
```

In Microsoft Excel for the Macintosh, the following macro formula formats the selected range of cells with the 10-point Courier bold italic font:

```
FORMAT.FONT("Courier",10,TRUE,TRUE,FALSE,FALSE,0,
FALSE,FALSE)
```

### Note

For macro compatibility with Microsoft Excel for the Macintosh, the presence of the *outline* and *shadow* arguments in the preceding formula would not prevent the macro from working on Microsoft Excel for Windows or OS/2 nor would their absence prevent it from working on the Macintosh.

### Related Functions

| | |
|---|---|
| ALIGNMENT | Aligns or wraps text in cells |
| FORMAT.NUMBER | Formats numbers, dates, and times |
| FORMAT.TEXT | Controls the alignment and orientation of the selected text |

*Macro Sheets Only*

## FORMAT.LEGEND(position_num)
## FORMAT.LEGEND?(position_num)

Equivalent to the Format Legend command. Determines the position and orientation of the legend on a chart and returns TRUE; returns an error message if the legend is not already selected. For detailed information on the Format Legend command, see online Help.

***Position_num*** is a number from 1 to 5 specifying the position of the legend.

| Position_num | Position of legend |
|---|---|
| 1 | Bottom |
| 2 | Corner |
| 3 | Top |
| 4 | Right |
| 5 | Left |

### Example

The following macro formula moves the selected legend to the bottom of the chart:

```
FORMAT.LEGEND(1)
```

### Related Functions

| | |
|---|---|
| FORMAT.MOVE | Moves the selected object |
| LEGEND | Adds or deletes a chart legend |

*Macro Sheets Only*

## FORMAT.MAIN(type_num,view,overlap, gap_width,vary,drop,hilo,angle,gap_depth,chart_depth)
## FORMAT.MAIN?(type_num,view,overlap, gap_width,vary,drop,hilo,angle,gap_depth,chart_depth)

Equivalent to the Format Main Chart command. Formats a chart according to the arguments you specify. For detailed information on the Format Main Chart command, see online Help.

***Type_num*** is a number from 1 to 10 specifying the type of chart.

| Type_num | Chart |
|---|---|
| 1 | Area |
| 2 | Bar |
| 3 | Column |
| 4 | Line |
| 5 | Pie |
| 6 | XY (Scatter) |
| 7 | 3-D Area |
| 8 | 3-D Column |
| 9 | 3-D Line |
| 10 | 3-D Pie |

***View*** is a number from 1 to 4 specifying one of the views in the Data View box in the Format Main Chart dialog box. The data views correspond to the chart *type_num* as shown in the following table. A blank entry in the table indicates that *view* 2, 3, or 4 is not available for that chart type.

| | View 1 | View 2 | View 3 | View 4 |
|---|---|---|---|---|
| **Area** | Over-lapped | Stacked | Stacked 100% | |
| **Bar** | Side-by-side | Stacked | Stacked 100% | |
| **Column** | Side-by-side | Stacked | Stacked 100% | |
| **Line** | Normal | Stacked | Stacked 100% | |
| **Pie** | Normal | | | |
| **XY (Scatter)** | Normal | | | |
| **3-D Area** | Stacked | Stacked 100% | 3-D layout | |
| **3-D Column** | Side-by-side | Stacked | Stacked 100% | 3-D layout |
| **3-D Line** | 3-D layout | | | |
| **3-D Pie** | Normal | | | |



***Overlap*** is a number from –100 to 100 specifying how you want bars or columns to be positioned. It corresponds to the Overlap box in the Format Main Chart dialog box. *Overlap* is ignored if *type_num* is not 2 or 3 (bar or column chart).

- If *overlap* is positive, it specifies the percentage of overlap you want for bars or columns. For example, 50 would cause one-half of a bar or column to be covered by an adjacent bar or column. A value of zero prevents bars or columns from overlapping.
- If *overlap* is negative, then bars or columns are separated by the specified percentage of the maximum available distance between any two bars or columns.
- If *overlap* is omitted, it is assumed to be 0, and bars or columns do not overlap.

***Gap_width*** is a number from 0 to 500 specifying the space between bar or column clusters as a percentage of the width of a bar or column. It corresponds to the Gap Width box in the Format Main Chart dialog box.

- *Gap_width* is ignored if *type_num* is not 2 or 3 (bar or column chart).
- If *gap_width* is omitted, it is assumed to be 50.

The next three arguments are logical values corresponding to check boxes in the Format Main Chart dialog box. If an argument is TRUE, Microsoft Excel turns on the corresponding check box; if FALSE, Microsoft Excel turns off the check box. If an argument is omitted, the setting is unchanged.

***Vary*** corresponds to the Vary By Categories check box. *Vary* applies only to charts with one data series and is not available for area charts.

***Drop*** corresponds to the Drop Lines check box. *Drop* is available only for area and line charts.

***Hilo*** corresponds to the Hi-Lo Lines check box. *Hilo* is available only for line charts.

***Angle*** is a number from 0 to 360 specifying the angle of the first pie slice (in degrees) if the chart is a pie chart. If *angle* is omitted, it is assumed to be 50.

The next two arguments are for 3-D charts only.

***Gap_depth*** is a number from 0 to 500 specifying the depth of the gap in front of and behind a bar, column, area, or line as a percentage of the depth of the bar, column, area, or line. *Gap_depth* corresponds to the Gap Depth box in the Format Main Chart dialog box.

- *Gap_depth* is ignored if the chart is a pie chart or if it is not a 3-D chart.
- If *gap_depth* is omitted and the view is 3-D layout, *gap_depth* is assumed to be 50. If *gap_depth* is omitted and the view is side-by-side, stacked, or stacked 100%, *gap_depth* is assumed to be 0.

***Chart_depth*** is a number from 20 to 2000 specifying the visual depth of the chart as a percentage of the width of the chart. *Chart_depth* corresponds to the Chart Depth box in the Format Main Chart dialog box.

- *Chart_depth* is ignored if the chart is not a 3-D chart.
- If *chart_depth* is omitted, it is assumed to be 100.

### Example

The following macro formula formats the main chart as a 3-D column chart with a side-by-side view, no overlap between the columns, 100% gap

depth, and 150% chart depth. All other arguments are default values.

FORMAT.MAIN(8,1,0,,,,,,100,150)

### Related Functions

| | |
|---|---|
| FORMAT.NUMBER | Formats numbers, dates, and times |
| FORMAT.OVERLAY | Formats an overlay chart |
| FORMAT.TEXT | Controls the alignment and orientation of the selected text |

Macro Sheets Only

## FORMAT.MOVE(*x_offset,y_offset,reference*)
## FORMAT.MOVE?(*x_offset,y_offset,reference*)

**Form 1: Worksheet objects**

Equivalent to moving an object with the mouse. Moves the selected object to the specified position and, if successful, returns TRUE. If the selected object cannot be moved, FORMAT.MOVE returns FALSE. There are two forms of this function. Use this form to move worksheet objects. Use the second form to move chart items. It is generally easier to use the macro recorder to enter this function on your macro sheet.

***X_offset*** specifies the horizontal position where you want to move the object and is measured in points from the upper-left corner of the object to the upper-left corner of the cell specified by *reference*. A point is 1/72nd of an inch.

***Y_offset*** specifies the vertical position where you want to move the object and is measured in points from the upper-left corner of the object to the upper-left corner of the cell specified by *reference*.

***Reference*** specifies which cell or range of cells to place the object in relation to.

- If *reference* is a range of cells, only the upper-left cell is used.
- If *reference* is omitted, it is assumed to be cell A1.

### Note

The position of an object is based on its upper-left corner. For ovals and arcs, the position is based on the upper-left corner of the bounding rectangle of the object.

### Example

The following macro formula moves an object on the active worksheet so that it is 10 points horizontally offset and 15 points vertically offset from cell D4:

FORMAT.MOVE(10,15,!$D$4)

### Related Functions

| | |
|---|---|
| CREATE.OBJECT | Creates the specified object |
| FORMAT.SIZE | Changes the size of the selected object |
| MOVE | Moves a window |

Macro Sheets Only

## FORMAT.MOVE(*x_pos,y_pos*)
## FORMAT.MOVE?(*x_pos,y_pos*)

**Form 2: Chart items**

Equivalent to moving an object with the mouse. Moves the base of the selected object to the position specified and, if successful, returns TRUE. If the selected object cannot be moved, FORMAT.MOVE returns FALSE. There are two forms of this function. Use this form to move chart items. Use the first form to move worksheet objects. It is generally easier to use the macro recorder to enter this function on your macro sheet.

***X_pos*** specifies the horizontal position where you want to move the object and is measured in points from the base of the object to the lower-left corner of the window. A point is 1/72nd of an inch.

***Y_pos*** specifies the horizontal position where you want to move the object and is measured in points from the base of the object to the lower-left corner of the window.



### Notes

- The base of a text label on a chart is the lower-left corner of the text rectangle.
- The base of an arrow is the end without the arrowhead.
- The base of a pie slice is the point.

### Example

On a chart, the following macro formula moves the base of the selected chart object 10 points to the right of and 20 points above the lower-left corner of the window:

FORMAT.MOVE(10,20)

### Related Functions

| | |
|---|---|
| FORMAT.SIZE | Changes the size of the selected object |
| MOVE | Moves a window |

Macro Sheets Only

## FORMAT.NUMBER(*format_text*)
## FORMAT.NUMBER?(*format_text*)

Equivalent to the Format Number command. Formats numbers, dates, and times in the selected cells. Use FORMAT.NUMBER to apply built-in formats or to create and apply custom formats.

***Format_text*** is a format string, such as "#,##0.00", specifying which format to apply to the selection.

For information on number and text formats, see Chapter 5, "Formatting a Worksheet," in the *Microsoft Excel User's Guide*.

### Example

The following macro formula applies a custom time format to the selected cells:

FORMAT.NUMBER("h"" hours ""m"" minutes ""s"" seconds""")

The preceding format would display 3:45:30 as "3 hours 45 minutes 30 seconds."

### Related Functions

| | |
|---|---|
| DELETE.FORMAT | Deletes the specified custom number format |
| FORMAT.FONT | Applies a font to the current selection |
| FORMAT.TEXT | Controls the alignment and orientation of the selected text |

Macro Sheets Only

## FORMAT.OVERLAY(*type_num,view,overlap,width,vary,drop,hilo,angle,series_dist,series_num*)
## FORMAT.OVERLAY?(*type_num,view,overlap,width,vary,drop,hilo,angle,series_dist,series_num*)

Equivalent to the Format Overlay command. Formats the overlay chart according to the arguments you specify. For detailed information on the Format Overlay command, see online Help.

***Type_num*** is a number from 1 to 6 specifying the type of chart.

| *Type_num* | Chart |
|---|---|
| 1 | Area |
| 2 | Bar |
| 3 | Column |
| 4 | Line |
| 5 | Pie |
| 6 | XY (Scatter) |

***View*** is a number from 1 to 3 specifying one of the views in the Data View box in the Format Overlay dialog box. The data views correspond to the chart *type_num* as shown in the following table. A blank entry in the table indicates that *view* 2 or 3 is not available for that chart type.

| | *View* 1 | *View* 2 | *View* 3 |
|---|---|---|---|
| **Area** | Overlapped | Stacked | Stacked 100% |
| **Bar** | Side-by-side | Stacked | Stacked 100% |
| **Column** | Side-by-side | Stacked | Stacked 100% |
| **Line** | Normal | Stacked | Stacked 100% |
| **Pie** | Normal | | |
| **XY (Scatter)** | Normal | | |

***Overlap*** is a number from –100 to 100 specifying how you want bars or columns to be positioned. It corresponds to the Overlap box in the Format Overlay dialog box. *Overlap* is ignored if *type_num* is not 2 or 3 (bar or column chart).

- If *overlap* is positive, it specifies the percentage of overlap you want for bars or columns. For example, 50 would cause one-half of a bar or column to be covered by an adjacent bar or column.
- If *overlap* is negative, then bars or columns are separated by the specified percentage of the maximum available distance between any two bars or columns.
- If *overlap* is omitted, it is assumed to be 0, and bars or columns do not overlap.

***Width*** is a number from 0 to 500 specifying the space between bar or column clusters as a percentage of the width of a bar or column. *Width* corresponds to the Gap Width box in the Format Overlay dialog box.

- *Width* is ignored if *type_num* is not 2 or 3 (bar or column chart).
- If *width* is omitted, it is assumed to be 50.

The next three arguments are logical values corresponding to check boxes in the Format Overlay dialog box. If an argument is TRUE, Microsoft Excel turns on the corresponding check box; if FALSE, Microsoft Excel turns off the check box. If an argument is omitted, the setting is unchanged.

***Vary*** corresponds to the Vary By Categories check box. *Vary* is not available for area charts.

***Drop*** corresponds to the Drop Lines check box. *Drop* is available only for area and line charts.

***Hilo*** corresponds to the Hi-Lo Lines check box. *Hilo* is available only for line charts.

***Angle*** is a number from 0 to 360 specifying the angle of the first pie slice (in degrees) if the chart is a pie chart. If *angle* is omitted, it is assumed to be 50.

***Series_dist*** is the number 1 or 2 and specifies automatic or manual series distribution.

- If *series_dist* is 1 or omitted, Microsoft Excel uses automatic series distribution.
- If *series_dist* is 2, Microsoft Excel uses manual series distribution, and you must specify which series is first in the distribution by using the *series_num* argument.

***Series_num*** is the number of the first series in the overlay chart and corresponds to the First Overlay Series box in the Format Overlay dialog box. If *series_dist* is 1 (automatic series distribution), this argument is ignored.

### Example

The following macro formula formats the overlay chart as a line chart with a normal view (*view* 1), hi-lo lines, and manual series distribution with series number 3 specified as the first series in the overlay. All other arguments are default values.

```
FORMAT.OVERLAY(4,1,,,,,TRUE,,2,3)
```

### Related Functions

| | |
|---|---|
| DELETE.OVERLAY | Deletes an overlay from a chart |
| FORMAT.MAIN | Formats the main chart |



*Macro Sheets Only*

## FORMAT.SIZE(*x_off,y_off,reference*)
## FORMAT.SIZE?(*x_off,y_off,reference*)

**Form 1: Worksheet objects, relative**

Equivalent to sizing an object with the mouse. Sizes the selected worksheet object and returns TRUE. If the selected object cannot be sized, FORMAT.SIZE returns FALSE. There are two forms of this function. Use this form to size worksheets objects relative to a cell or range of cells. Use the second form to size worksheet objects and chart items. It is generally easier to use the macro recorder to enter this function on your macro sheet.

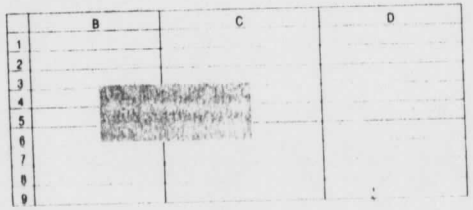
***X_off*** specifies the width of the selected object and is measured in points from the lower-right corner of the object to the upper-left corner of *reference*. A point is 1/72nd of an inch. If omitted, *x_off* is assumed to be 0.

***Y_off*** specifies the height of the selected object and is measured in points from the lower-right corner of the object to the upper-left corner of *reference*. If omitted, *y_off* is assumed to be 0.

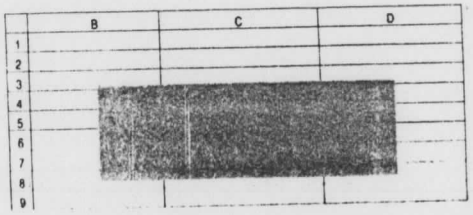
***Reference*** specifies the cell or range of cells to use as the basis for the offset and for sizing. If *reference* is a range of cells, only the upper-left cell in the range is used.

### Examples

Suppose you want to change the size of an object from this size:




To this size:




In the second illustration, the right edge of the object is about 34 points offset from the left edge of cell D7 (if it were 0 points offset, the right edge of the object would be directly over the left edge of column D). The bottom edge of the object is offset about 6 points from the upper edge of cell D7.

The following macro formula sizes the object approximately as shown:

```
FORMAT.SIZE(34,6,D7)
```

### Related Functions

| | |
|---|---|
| FORMAT.MOVE | Moves the selected object |
| SIZE | Changes the size of a window |

*Macro Sheets Only*

## FORMAT.SIZE(*width,height*)
## FORMAT.SIZE?(*width,height*)

**Form 2: Worksheet objects and chart items, absolute**

Equivalent to sizing an object with the mouse. Sizes the selected object and returns TRUE. If the selected chart object cannot be sized, FORMAT.SIZE returns FALSE. There are two forms of this function. Use this form to size worksheet objects and chart items absolutely. Use the first form to size only worksheet objects. It is generally easier to use the macro recorder to enter this function on your macro sheet.

***Width*** specifies the width of the selected chart object, measured in points. A point is 1/72nd of an inch.

***Height*** specifies the height of the selected chart object, measured in points.

You do not always have to use both arguments. For example, if you specify height and not width, the height changes but the width does not.

### Notes

- The base of a text label on a chart is the lower-left corner of the text rectangle.
- The base of an arrow is the end without the arrowhead.

### Example

The following macro formula sizes the selected object to 175 × 111 points:

```
FORMAT.SIZE(175,111)
```

### Related Functions

| | |
|---|---|
| FORMAT.MOVE | Moves the selected object |
| SIZE | Changes the size of a window |

Macro Sheets Only

## FORMAT.TEXT(*x_align,y_align,orient_num, auto_text,auto_size,show_key,show_value*)

## FORMAT.TEXT?(*x_align,y_align,orient_num, auto_text,auto_size,show_key,show_value*)

Equivalent to the Format Text command. Formats the selected worksheet text box or any text item on a chart. For detailed information on the Format Text command, see online Help.

Arguments correspond to check boxes or options in the Format Text dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***X_align*** is a number from 1 to 3 specifying the horizontal alignment of the text.

| *X_align* | Horizontal alignment |
|---|---|
| 1 | Left |
| 2 | Center |
| 3 | Right |

***Y_align*** is a number from 1 to 3 specifying the vertical alignment of the text.

| *Y_align* | Vertical alignment |
|---|---|
| 1 | Top |
| 2 | Center |
| 3 | Bottom |

***Orient_num*** is a number from 0 to 3 specifying the orientation of the text.

| *Orient_num* | Arranges the text |
|---|---|
| 0 | Horizontally |
| 1 | Vertically |
| 2 | Upward |
| 3 | Downward |

***Auto_text*** corresponds to the Automatic Text check box. If the selected text was created with the Chart Attach Text command and later edited, this option restores the original text. *Auto_text* is ignored for text boxes on worksheets and macro sheets.

***Auto_size*** corresponds to the Automatic Size check box. If you have changed the size of the border around the selected text, this option restores the border to automatic size. Automatic size makes the border fit exactly around the text no matter how you change the text.

***Show_key*** corresponds to the Show Key check box. This argument applies only if the selected text is an attached data label on a chart.

***Show_value*** corresponds to the Show Value check box. This argument applies only if the selected text is an attached data label on a chart.

The following list summarizes which arguments apply to each type of text item:



| Text Item | Arguments that apply |
|---|---|
| Worksheet text box | *X_align, y_align, orient_num, auto_size* |
| Attached data label | All arguments |
| Unattached text label | *X_align, y_align, orient_num, auto_size* |
| Tickmark label | *Orient_num* |

### Example

The following macro formula formats the selected text so that its horizontal alignment is to the right, its vertical alignment is to the bottom, its orientation is downward, the automatic text option is off, and the automatic size option is on. Assume that the text is not an attached data label.

```
FORMAT.TEXT(3,3,3,FALSE,TRUE,,)
```

### Related Functions

| | |
|---|---|
| CREATE.OBJECT | Creates a specified object, such as a text box |
| FORMAT.FONT | Applies a font to the current selection |
| FORMULA | Enters a formula in a cell or text on a chart |

Macro Sheets Only

## FORMULA(*formula_text,reference*)

Enters a formula in the active cell or in a reference. Use FORMULA to enter numbers, text, references, and formulas in a Microsoft Excel document.

***Formula_text*** can be text, or a number, reference, or formula in the form of text, or a reference to a cell containing any of the above. Certain restrictions (described below) apply to FORMULA when *reference* refers to a chart or, if *reference* is omitted, when the active document is a chart.

***Reference*** specifies where *formula_text* is to be entered. It can be a reference to a cell in the active worksheet or an external reference to a worksheet. On a chart, *reference* can be either a text label or a series formula. If *reference* is omitted, *formula_text* is entered in the active cell.

If the active document is a worksheet, using FORMULA is equivalent to entering *formula_text* into the cell specified by *reference*. *Formula_text* is entered just as if you typed it in the formula bar.

- If *formula_text* is a formula, the formula is entered. Text arguments must be surrounded by double sets of quotation marks. For example, to enter the formula =ALERT("Hello World",1) in the active cell with the FORMULA function, you would use the formula
  `FORMULA("=ALERT(""Hello World"",1)")`
- If *formula_text* is a number, text, or logical value, the value is entered as a constant.
- If *formula_text* contains references, those references must be R1C1-style references, such as "=RC[1]*(1+R1C1)". If you are recording a macro when you enter a formula, Microsoft Excel converts A1-style references to R1C1-style references. For example, if you enter the formula =B2*(1+$A$1) in cell C2 while recording, Microsoft Excel records that action as =FORMULA("=RC[−1]*(1+R1C1)").

If the active document is a chart, using FORMULA is equivalent to entering a text label or a SERIES formula.

| If | Then |
|---|---|
| *Formula_text* can be treated as a text label, and the current selection is a text label | The selected text label is replaced with *formula_text*. |
| *Formula_text* can be treated as a text label, and there is no current selection or the current selection is not a text label | *Formula_text* makes a new text label. |
| *Formula_text* can be treated as a SERIES formula, and the current selection is a SERIES formula | The selected SERIES formula is replaced with *formula_text*. |
| *Formula_text* can be treated as a SERIES formula, and the current selection is not a SERIES formula | *Formula_text* makes a new SERIES formula. |

### Examples

If the active document is a worksheet, the following macro formula enters the constant number 523 in the active cell:

```
FORMULA(523)
```

If the active document is a worksheet and the active cell is C2, the following macro formula enters the formula =RC[−1]*(1+R1C1) in cell C2:

```
FORMULA("=RC[-1]*(1+R1C1)")
```

If the active document is a worksheet, the following macro formula enters the result of the INPUT function in cell A5:

```
FORMULA(INPUT("Enter a formula:",0),"R5C1")
```

If the active document is a worksheet, the following macro formula enters the number 1000 in the cell two rows down and three columns right from the active cell:

```
FORMULA(1000,"R[2]C[3]")
```

The following macro formula enters the phrase "Year to Date" in cell B4 on the sheet named Sales:

```
FORMULA("Year to Date",Sales!B4)
```

If the active document is a chart, the following macro formula enters a SERIES formula on the chart. If the current selection is a SERIES formula, it is replaced:

```
FORMULA("=SERIES(""Chart Title"",,{1,2,3},1)")
```

### Related Functions

| | |
|---|---|
| FORMULA.ARRAY | Enters an array formula on a document |
| FORMULA.FILL | Fills a range with a formula |
| SET.VALUE | Sets the value of a cell on the macro sheet |

*Macro Sheets Only*

## FORMULA.ARRAY(*formula_text,reference*)

Enters a formula as an array formula in the range specified or in the current selection. Equivalent to entering an array formula while pressing CTRL+SHIFT+ENTER in Microsoft Excel for Windows or OS/2 or COMMAND+ENTER in Microsoft Excel for the Macintosh.

***Formula_text*** is the text you want to enter in the array. For more information on *formula_text*, see FORMULA.

***Reference*** specifies where *formula_text* is entered. It can be a reference to a cell on the active worksheet or an external reference to a named document. *Reference* must be a quoted R1C1-style reference. If *reference* is omitted, *formula_text* is entered in the active cell.

### Examples

If the selection is D25:E25, the following macro formula enters the array formula {=D22:E22+D23:E23} in the range D25:E25:

```
FORMULA.ARRAY("=R[-3]C:R[-3]C[1]+R[-2]C:R[-2]C[1]")
```

Regardless of the selection, the following macro formula enters the array formula {=D22:E22+D23:E23} in the range D25:E25:

```
FORMULA.ARRAY("=R[-3]C:R[-3]C[1]
+R[-2]C:R[-2]C[1]","R25C4:R25C5")
```

To use FORMULA.ARRAY to put an array in a specific document, specify the name of the document as an external reference in the *reference* argument. In Microsoft Excel for Windows or OS/2, using "SALES.XLS!R25C3:R25C4" as the *reference* argument in the preceding example would enter the array in cells C25:D25 on the worksheet named SALES.XLS. In Microsoft Excel for the Macintosh, using "Sales!R25C3:R25C4" as the *reference* argument would enter the array in the same cells in the worksheet named Sales.

### Related Functions

| | |
|---|---|
| FORMULA | Enters a formula in a cell or text on a chart |
| FORMULA.FILL | Fills a range with a formula |



*Macro Sheets Only*

## FORMULA.CONVERT(*formula_text,from_a1, to_a1,to_ref_type,rel_to_ref*)

Changes the reference style and type of a formula between A1 and R1C1 and between relative and absolute. Use FORMULA.CONVERT to convert references of one style or type to another style or type.

***Formula_text*** is the formula, given as text, containing the references you want to change. *Formula_text* must be a valid formula and an equal sign must be included.

***From_a1*** is a logical value specifying whether the references in *formula_text* are in A1 or R1C1 style. If *from_a1* is TRUE, references are in A1 style; if FALSE, references are in R1C1 style.

***To_a1*** is a logical value specifying the form for the references FORMULA.CONVERT returns. If *to_a1* is TRUE, references are returned in A1 style; if FALSE, references are returned in R1C1 style. If *to_a1* is omitted, the reference style is not changed.

***To ref type*** is a number from 1 to 4 specifying the reference type of the returned formula. If *to_ref_type* is omitted, the reference type is not changed.

| *To_ref_type* | Reference type returned |
|---|---|
| 1 | Absolute |
| 2 | Absolute row, relative column |
| 3 | Relative row, absolute column |
| 4 | Relative |

***Rel_to_ref*** is an absolute reference that specifies what cell the relative references are or should be relative to.

### Examples

A typical use for FORMULA.CONVERT is to convert relative references entered by the user in an INPUT or custom dialog box into absolute references. The following macro formula converts the given formula to an absolute, R1C1-style reference:

```
FORMULA.CONVERT("=A1:A10",TRUE,FALSE,1) equals
"=R1C1:R10C1"
```

The following macro formula converts the references in the given formula to relative, A1-style references:

```
FORMULA.CONVERT("=SUM(R10C2:R15C2)",FALSE,TRUE,
4) equals "=SUM(B10:B15)"
```

### Tip

To put the converted formula into a cell or range of cells, use the FORMULA.CONVERT function as the *formula_text* argument to the FORMULA function.

### Related Functions

| | |
|---|---|
| ABSREF | Returns the absolute reference of a cell |
| RELREF | Returns the relative reference of a cell |

*Macro Sheets Only*

## FORMULA.FILL(*formula_text,reference*)

Enters a formula in the range specified or in the current selection. Equivalent to entering a formula in a range of cells while pressing CTRL in Microsoft Excel for Windows or OS/2, or OPTION in Microsoft Excel for the Macintosh.

***Formula text*** is the text with which you want to fill the range. For more information on *formula_text*, see FORMULA.

***Reference*** is optional and specifies where *formula_text* is entered. It can be a reference to a range in the active worksheet or an external reference to a named document.

### Examples

If the active range is D25:E25, the following macro formula fills the formula SUM(D15:D24) in cell D25 and SUM(E15:E24) in cell E25:

```
FORMULA.FILL("=SUM(R[-1]C:R[-10]C)")
```

The following macro formula puts the same formula in the same cells as the first example, no matter what the active cell or range is:

```
FORMULA.FILL("=SUM(R[-1]C:R[-10]C)","R25C4:R25C5")
```

In Microsoft Excel for Windows or OS/2, the following macro formula also puts the same formula in the same cells as the first example, on the worksheet named SALES.XLS:

```
FORMULA.FILL("=SUM(R[-1]C:R[-10]C)",
"SALES.XLS!R25C4:R25C5")
```

In Microsoft Excel for the Macintosh, the following macro formula also puts the same formula in the same cells as the first example, on the worksheet names Sales:

```
FORMULA.FILL("=SUM(R[-1]C:R[-10]C)",
"Sales!R25C4:R25C5")
```

Also see the last example in REFTEXT.

### Related Functions

| | |
|---|---|
| FORMULA | Enters a formula in a cell or text on a chart |
| FORMULA.ARRAY | Enters an array formula on a document |

Macro Sheets Only

## FORMULA.FIND(*text,in_num,at_num,by_num, dir_num,match_case*)
## FORMULA.FIND?(*text,in_num,at_num,by_num, dir_num,match_case*)

Equivalent to the Formula Find command. Selects the next or previous cell containing the specified text and returns TRUE. If a matching cell is not found, FORMULA.FIND returns FALSE and displays a message. For detailed information on the Formula Find command, see online Help.

***Text*** is the text you want to find. *Text* corresponds to the Find What box in the Formula Find dialog box.

***In_num*** is a number from 1 to 3 specifying where to search.

| *In_num* | Searches |
|---|---|
| 1 | Formulas |
| 2 | Values |
| 3 | Notes |

***At_num*** is the number 1 or 2 and specifies whether to find cells containing only *text* or also cells containing *text* within a longer string of characters.

| *At_num* | Searches for *text* as |
|---|---|
| 1 | A whole string (the only value in the cell) |
| 2 | Either a whole string or part of a longer string |

***By_num*** is the number 1 or 2 and specifies whether to search by rows or by columns.

| *By_num* | Searches by |
|---|---|
| 1 | Rows |
| 2 | Columns |

***Dir_num*** is the number 1 or 2 and specifies whether to search for the next or previous occurrence of *text*.

| *Dir_num* | Searches for |
|---|---|
| 1 or omitted | The next occurrence of *text* |
| 2 | The previous occurrence of *text* |

***Match_case*** is a logical value corresponding to the Match Case check box in the Formula Find dialog box. If *match_case* is TRUE, Microsoft Excel matches characters exactly, including uppercase and lowercase; if FALSE or omitted, matching is not case sensitive.

### Notes

- In Microsoft Excel for Windows or OS/2, the dialog-box form of FORMULA.FIND is equivalent to pressing SHIFT+F5.
- If more than one cell is selected when you use FORMULA.FIND, Microsoft Excel searches only that selection.

### Example

The following macro formula searches by columns for the next occurrence of "units per month":

```
FORMULA.FIND("units per month",2,2,2,1)
```

Also see the third example for MAX.



Macro Sheets Only

## FORMULA.FIND.NEXT()
## FORMULA.FIND.PREV()

Equivalent to pressing F7 and SHIFT+F7 in Microsoft Excel for Windows or OS/2, or COMMAND+H and COMMAND+SHIFT+H in Microsoft Excel for the Macintosh. These functions find the next and previous cells on the worksheet, as specified in the Formula Find dialog box, and return TRUE. If a matching cell is not found, the functions return FALSE. For more information see FORMULA.FIND.

### Related Function

| | |
|---|---|
| FORMULA.FIND | Finds text in a document |

Macro Sheets Only

## FORMULA.GOTO(*reference,corner*)
## FORMULA.GOTO?(*reference,corner*)

Equivalent to the Formula Goto command or to pressing F5. Scrolls through the worksheet and selects a named area or reference. For detailed information on the Formula Goto command, see online Help.

***Reference*** specifies where to scroll and what to select.

- *Reference* should be either an external reference to a document, an R1C1-style reference in the form of text (see the first example below), or a name.
- If the Formula Goto command has already been carried out, *reference* is optional. If *reference* is omitted, it is assumed to be the reference of the cells you selected before the previous Formula Goto command or FORMULA.GOTO macro function was carried out. This feature distinguishes FORMULA.GOTO from SELECT.

***Corner*** is a logical value that specifies whether to scroll the window so that the upper-left cell in *reference* is in the upper-left corner of the active window. If *corner* is TRUE, Microsoft Excel places *reference* in the upper-left corner of the window; if FALSE or omitted, Microsoft Excel scrolls normally.

### Note

If you are recording a macro, the reference you enter in the Reference box of the Formula Goto dialog box is recorded as text in the R1C1 reference style.

### Examples

Each of the following macro formulas goes to cell A1 on the active worksheet:

```
FORMULA.GOTO(!$A$1)
FORMULA.GOTO("R1C1")
```

Each of the following macro formulas goes to the cells named Sales on the active worksheet and scrolls so that the upper-left corner of Sales is in the upper-left corner of the window:

```
FORMULA.GOTO(!Sales,TRUE)
FORMULA.GOTO("Sales",TRUE)
```

In Microsoft Excel for Windows or OS/2, each of the following macro formulas goes to cells B2:C3 on a worksheet named BUDGET.XLS:

```
FORMULA.GOTO(BUDGET.XLS!$B$2:$C$3)
FORMULA.GOTO("BUDGET.XLS!R2C2:R3C3")
```

In Microsoft Excel for the Macintosh, each of the following macro formulas goes to cells B2:C3 on a worksheet named Budget:

```
FORMULA.GOTO(Budget!$B$2:$C$3)
FORMULA.GOTO("Budget!R2C2:R3C3")
```

### Related Functions

| | |
|---|---|
| HSCROLL or VSCROLL | Horizontally or vertically scrolls a document by percentage or by column or row number |
| SELECT | Selects cells on a worksheet or an item on a chart |

Macro Sheets Only

## FORMULA.REPLACE(find_text,replace_text, look_at,look_by,active_cell,match_case)

## FORMULA.REPLACE?(find_text, replace_text,look_at,look_by,active_cell,match_case)

Equivalent to the Formula Replace command (full menus). Finds and replaces characters in cells on your worksheet. For detailed information on the Formula Replace command, see online Help.

***Find_text*** is the text you want to find. You can use the wildcard characters, question mark (?) and asterisk (*), in *find_text*. A question mark matches any single character; an asterisk matches any sequence of characters. If you want to find an actual question mark or asterisk, type a tilde (~) before the character.

***Replace_text*** is the text you want to replace *find_text* with.

***Look_at*** is a number specifying whether you want *find_text* to match the entire contents of a cell or any string of matching characters:

| *Look_at* | Looks for *find_text* |
|---|---|
| 1 | As the entire contents of a cell |
| 2 or omitted | As part of the contents of a cell |

***Look_by*** is a number specifying whether to search horizontally (through rows) or vertically (through columns):

| *Look_by* | Looks for *find_text* |
|---|---|
| 1 or omitted | By rows |
| 2 | By columns |

***Active_cell*** is a logical value specifying the cells in which *find_text* is to be replaced.

- If *active_cell* is TRUE, *find_text* is replaced in the active cell only.
- If *active_cell* is FALSE or omitted, *find_text* is replaced in the entire selection, or, if the selection is a single cell, in the entire document.

***Match_case*** is a logical value corresponding to the Match Case check box in the Formula Replace dialog box. If *match_case* is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box. If *match_case* is omitted, the status of the check box is unchanged.

### Notes

- In FORMULA.REPLACE?, the dialog-box form of the function, omitted arguments are assumed to be the same arguments used in the previous replace operation. If there was no previous replace operation, omitted text arguments are assumed to be "" (empty text).
- The result of FORMULA.REPLACE must be a valid cell entry. For example, you cannot replace "=" with "==" at the beginning of a formula.
- If more than a single cell is selected before you use FORMULA.REPLACE, only the selected cells are searched.

### Examples

The following macro formula replaces 7 with 8 in the entire selection, or in the entire document if the selection is a single cell:

```
FORMULA.REPLACE("7","8")
```

The following macro formula searches the selection and replaces any instances of "y" followed by any two characters, with just "y":

```
FORMULA.REPLACE("y??","y")
```

The following macro formula searches the selection and replaces any instances of "y" followed by any characters, with just "y":

```
FORMULA.REPLACE("y*","y")
```

The following macro formula searches the active cell and replaces any instances of "y" followed by two question marks, with just "y":

```
FORMULA.REPLACE("y~?~?","y",,,TRUE,)
```

### Related Functions

| | |
|---|---|
| FORMULA.FIND | Finds text in a document |
| REPLACE | Replaces characters in a string of text |



Macro Sheets Only

## FPOS(file_num,position_num)

Sets the position of a file. The position of a file is where a character is read from or written to by an FREAD, FREADLN, FWRITE, or FWRITELN function. Use FPOS when you want to write characters to or read characters from specific locations. For example, to append text to the end of a file, you must set the position to the end of the file; otherwise, you might accidentally overwrite existing characters in the file.

***File_num*** is the unique ID number of the file for which you want to set the position. *File_num* is returned by a previously executed FOPEN function. If *file_num* is not valid, FPOS returns the #VALUE! error value.

***Position_num*** is the location in the file that a character will be read from or written to.

- The first position in a file is 1—the location of the first character.
- The last position in the file is the same as the value returned by FSIZE. For example, the last position in a file with 280 characters is 280.
- If *position_num* is omitted, FPOS returns the current position of the file—that is, the number corresponding to where the next character will be read from or written to.

Whenever you read a character from or write a character to a file, the file's position is automatically incremented.

### Examples

The following macro appends to a text file the contents of the current selection on the active worksheet. FSIZE returns the number of characters in the file, which is the same as the position of the last character, and the macro writes cell contents, followed by a carriage return, to that location.

| | A | B | C |
|---|---|---|---|
| 11 | *Names* | *Formulas* | *Comments* |
| 12 | | | |
| 13 | | AppendTextToFile | Adds current selection to file |
| 14 | FileName | =INPUT("Name of file to append:",2) | |
| 15 | FileNumber | =FOPEN(FileName,1) | Opens the file as read/write |
| 16 | LastChar | =FSIZE(FileNumber) | Gets last char number |
| 17 | | =FPOS(FileNumber,(LastChar+1)) | Positions file after last char |
| 18 | | =FOR.CELL("CurrentCell") | |
| 19 | | =FWRITELN(FileNumber,CurrentCell) | Writes cell contents to file |
| 20 | | =NEXT() | |
| 21 | | =FCLOSE(FileNumber) | |
| 22 | | =RETURN() | |

Suppose you have a file created by another application, where the first five characters in that file are special codes that have no meaning in Microsoft Excel. With the FPOS function, you can skip the first five characters and then read the rest of the file using the FREADLN function.

| | A | B | C |
|---|---|---|---|
| 25 | *Names* | *Formulas* | *Comments* |
| 26 | | | |
| 27 | | SkipFiveChars | Skips characters, reads file |
| 28 | FileName | =INPUT("Name of file to read:",2) | |
| 29 | FileNumber | =FOPEN(FileName,2) | Opens file as read-only |
| 30 | | =FPOS(FileNumber,6) | Skips first five characters |
| 31 | | =WHILE(FPOS(FileNumber)<=FSIZE(FileNu | Reads until end of file |
| 32 | ReadLine | =FREADLN(FileNumber) | |
| 33 | | =FORMULA(ReadLine) | Places line in active cell |
| 34 | | =SELECT(OFFSET(ACTIVE.CELL(),1,0)) | Moves active cell to next row |
| 35 | | =NEXT() | |
| 36 | | =FCLOSE(FileNumber) | |
| 37 | | =RETURN() | |

The complete formula in cell B31 is:

```
=WHILE(FPOS(FileNumber)<=FSIZE(FileNumber))
```

Also see the example in RESULT.

### Related Functions

| | |
|---|---|
| FCLOSE | Closes a text file |
| FOPEN | Opens a text file |
| FREAD | Reads characters from a text file |
| FREADLN | Reads a line from a text file |
| FWRITE | Writes characters to a text file |
| FWRITELN | Writes a line to a text file |

*Macro Sheets Only*

## FREAD(file_num,num_chars)

Reads characters from a file, starting at the current position in the file. (For more information about a file's position, see FPOS.) If FREAD is successful, it returns the text to the cell containing FREAD. If the end of the file is reached or if FREAD can't read the file, it returns the #N/A error value. Use FREAD instead of FREADLN when you need to read a specific number of characters from a text file.

***File_num*** is the unique ID number of the file you want to read data from. *File_num* is returned by a previously executed FOPEN function. If *file_num* is not valid, FREAD returns the #VALUE! error value.

***Num_chars*** specifies how many characters to read from the file. FREAD can read up to 255 characters at a time.

### Example

The following macro reads a file and displays its characters along with their code numbers. In each row of the active worksheet, the macro places 16 decimal codes in the first 16 cells; it places the corresponding 16 characters in the 17th cell. This is similar to some debugging or memory "dump" programs that show in rows the hexadecimal codes for 16 bytes and the 16 ASCII characters corresponding to those codes (some codes cannot be displayed because they do not correspond to printable characters). This macro assumes that the correct external worksheet is active when you run it.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **Main Program** | Characters to codes |
| 4 | | **DisplayCharacterCodes** | |
| 5 | | =OpenFile() | Opens the file |
| 6 | | =SetUpScreen() | Sets up the screen |
| 7 | | =ReadCharacters() | Reads chars and codes |
| 8 | | =RETURN() | |
| 9 | | | |
| 10 | | | |
| 11 | | **OpenFile** | Subroutine to open the file |
| 12 | | =ERROR(FALSE) | Disables error messages |
| 13 | FileName | =INPUT("Name of file to read",2) | Asks for the filename |
| 14 | FileNumber | =FOPEN(FileName,2) | Tries to open the file |
| 15 | | =IF(ISERROR(FileNumber)) | If file not there, displays |
| 16 | | =ALERT("File not available",2) | error message |
| 17 | | =HALT() | and halts the macro |
| 18 | | =END.IF() | |
| 19 | | =ERROR(TRUE) | Enables error messages |
| 20 | | =RETURN() | |
| 21 | | | |
| 22 | | | |
| 23 | | **SetUpScreen** | Sets up the screen |
| 24 | | =SELECT(!$A:$P) | Selects the first 16 columns |
| 25 | | =COLUMN.WIDTH(3) | Narrows the columns |
| 26 | | =SELECT(!$A$1) | Selects the upper-left cell |
| 27 | | =RETURN() | |
| 28 | | | |
| 29 | | | |
| 30 | | **ReadCharacters** | Reads and displays chars |
| 31 | | =ECHO(FALSE) | Turns off screen updating |
| 32 | LastChar | =FSIZE(FileNumber) | Position of last character |
| 33 | | =SET.NAME("TextString","") | Initializes text variable |
| 34 | | =FOR("Counter",1,LastChar) | Loops for each character |
| 35 | | =IF(AND(Counter<>1,MOD(Counter-1,16)=0 | Have 16 chars been read? |
| 36 | | =FORMULA(TextString) | Places 16 chars in col 17 |
| 37 | | =SET.NAME("TextString","") | Re-initializes text variable |
| 38 | | =SELECT(OFFSET(ACTIVE.CELL(),1,-16)) | Moves to next cell |
| 39 | | =END.IF() | |
| 40 | NextChar | =FREAD(FileNumber,1) | Reads the next character |
| 41 | | =FORMULA(CODE(NextChar)) | Puts char in the active cell |
| 42 | | =SET.NAME("TextString",TextString&NextCh | Adds char to variable |
| 43 | | =SELECT(OFFSET(ACTIVE.CELL(),0,1)) | Selects the next cell |
| 44 | | =NEXT() | |
| 45 | | =IF(MOD(Counter,16)<>0,FORMULA(TextSt | Puts text variable in col 17 |
| 46 | | =RETURN() | |

The complete formulas in cells B35, B42, and B45 are:

```
=IF(AND(Counter<>1,MOD(Counter-1,16)=0))
```

```
=SET.NAME("TextString",TextString&NextChar)
```

```
=IF(MOD(Counter,16)<>0,FORMULA(TextString,"RC17"))
```



### Related Functions

| | |
|---|---|
| FOPEN | Opens a text file |
| FREADLN | Reads a line from a text file |
| FWRITE | Writes characters to a text file |

*Macro Sheets Only*

## FREADLN(file_num)

Reads characters from a file, starting at the current position in the file and continuing to the end of the line, placing the characters in the cell containing FREADLN. (For more information about a file's position, see FPOS.) If FREADLN is successful, it returns the text it read, up to but not including the carriage-return and linefeed characters at the end of the line (in Microsoft Excel for Windows or OS/2) or the carriage-return character at the end of the line (in Microsoft Excel for the Macintosh). If the current file position is the end of the file or if FREADLN can't read the file, it returns the #N/A error value.

***File_num*** is the unique ID number of the file you want to read data from. *File_num* is returned by a previously executed FOPEN function. If *file_num* is not valid, FREADLN returns the #VALUE! error value.

### Example

See the second example in FPOS.

### Related Functions

| | |
|---|---|
| FREAD | Reads characters from a text file |
| FWRITE | Writes characters to a text file |
| FWRITELN | Writes a line to a text file |

*Macro Sheets Only*

## FREEZE.PANES(logical)

Equivalent to the Options Freeze Panes and Options Unfreeze Panes commands (full menus). Freezes or unfreezes panes that were split by a previously executed SPLIT function. Returns the #VALUE! error value if no panes are set or if the active window is not a worksheet or macro sheet. Use FREEZE.PANES to keep row or column titles on the screen while scrolling to other parts of the worksheet. For detailed information on the Options Freeze Panes and Options Unfreeze Panes commands, see online Help.

***Logical*** is a logical value specifying which command FREEZE.PANES is equivalent to.

- If *logical* is TRUE, the function is equivalent to the Options Freeze Panes command.
- If *logical* is FALSE, the function is equivalent to the Options Unfreeze Panes command.

### Example

The following macro freezes the first row and column so that you can still see the column and row headings when you scroll:

| | A | B | C |
|---|---|---|---|
| 11 | *Names* | *Formulas* | *Comments* |
| 12 | | | |
| 13 | | **SplitAndFreeze** | Splits sheet, freezes pane |
| 14 | | =SPLIT(,1) | Splits horizontally |
| 15 | | =SPLIT(1,) | Splits vertically |
| 16 | | =FREEZE.PANES(TRUE) | Freezes first row and col |
| 17 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| ACTIVATE | Activates a window or a pane of a window |
| SPLIT | Splits the active window into panes |

Macro Sheets Only

## FSIZE(file_num)

Returns the number of characters in a file. Use FSIZE to determine the size of the file, which is the same as the position of the last character in the file.

***File_num*** is the unique ID number of the file whose size you want to know. *File_num* is returned by a previously executed FOPEN function. If *file_num* is not valid, FSIZE returns the #VALUE! error value.

### Example

See the examples in FPOS and FREAD.

### Related Functions

| | |
|---|---|
| FOPEN | Opens a text file |
| FPOS | Returns position in a text file |

Macro Sheets Only

## FULL(logical)

Changes the active window from its normal size to full size or from full size to its previous size. In Microsoft Excel for Windows or OS/2, using FULL is equivalent to pressing CTRL+F10 (full size) and CTRL+F5 (previous size) or double-clicking the title bar. In Microsoft Excel for the Macintosh, using FULL is equivalent to double-clicking the title bar or clicking the zoom box.

***Logical*** is a logical value specifying the size of the window.

- If *logical* is TRUE, Microsoft Excel enlarges the window to its full size.
- If *logical* is FALSE, Microsoft Excel restores the window to its previous size.
- If *logical* is omitted and the window is not already enlarged, Microsoft Excel enlarges it. If *logical* is omitted and the window is enlarged, Microsoft Excel restores it to its previous size.

### Example

The following macro formula enlarges the active window to full size:

```
FULL(TRUE)
```

### Related Functions

| | |
|---|---|
| FORMAT.SIZE | Changes the size of the selected object |
| SIZE | Changes the size of a window |

## FV(rate,nper,pmt,pv,type)

Returns the future value of an investment based on periodic, constant payments and a constant interest rate.

For a more complete description of the arguments in FV and for more information on annuity functions, see PV.

***Rate*** is the interest rate per period.

***Nper*** is the total number of payment periods in an annuity.

***Pmt*** is the payment made each period; it cannot change over the life of the annuity. Typically, *pmt* contains principal and interest but no other fees or taxes.

***Pv*** is the present value, or the lump-sum amount that a series of future payments is worth right now. If *pv* is omitted, it is assumed to be 0.

***Type*** is the number 0 or 1 and indicates when payments are due. If *type* is omitted, it is assumed to be 0.

| Set *type* equal to | If payments are due |
|---|---|
| 0 | At the end of the period |
| 1 | At the beginning of the period |



### Notes

- Make sure that you are consistent about the units you use for specifying *rate* and *nper*. If you make monthly payments on a four-year loan at 12% annual interest, use 12%/12 for *rate* and 4*12 for *nper*. If you make annual payments on the same loan, use 12% for *rate* and 4 for *nper*.
- For all the arguments, cash you pay out, such as deposits to savings, is represented by negative numbers; cash you receive, such as dividend checks, is represented by positive numbers.

### Examples

FV(0.5%,10,–200,–500,1) equals $2,581.40

FV(1%,12,–1000) equals $12,682.50

FV(11%/12,35,–2000,,1) equals $82,846.25

Suppose you want to save money for a special project occurring a year from now. You deposit $1,000 into a savings account that earns 6% annual interest compounded monthly (monthly interest of 6%/12, or 0.5%). You plan to deposit $100 at the beginning of every month for the next 12 months. How much money will be in the account at the end of 12 months?

FV(0.5%,12,–100,–1000,1) equals $2,301.40

### Related Functions

| | |
|---|---|
| IPMT | Returns the interest payment for an investment for a given period |
| NPER | Returns the number of payments for an investment |
| PMT | Returns the periodic total payment for an investment |
| PPMT | Returns the payment on the principal for an investment for a given period |
| PV | Returns the present value of an investment |
| RATE | Returns the interest rate per period of an investment |

Macro Sheets Only

## FWRITE(file_num,text)

Writes text to a file, starting at the current position in that file. (For more information about a file's position, see FPOS.) If FWRITE can't write to the file, it returns the #N/A error value.

***File_num*** is the unique ID number of the file you want to write data to. *File_num* is returned by a previously executed FOPEN function. If *file_num* is not valid, FWRITE returns the #VALUE! error value.

***Text*** is the text you want to write to the file.

### Example

See FOPEN.

### Related Functions

| | |
|---|---|
| FOPEN | Opens a text file |
| FREAD | Reads characters from a text file |
| FWRITELN | Writes a line to a text file |

Macro Sheets Only

## FWRITELN(file_num,text)

Writes text, followed by a carriage return and linefeed, to a file, starting at the current position in that file. (For more information about a file's position, see FPOS.) If FWRITELN can't write to the file, it returns the #N/A error value. Use FWRITELN instead of FWRITE when you want to append a carriage return and linefeed to each group of characters that you write to a text file.

### Note

In Microsoft Excel for the Macintosh, FWRITELN writes text followed by a carriage return only.

***File_num*** is the unique ID number of the file you want to write data to. *File_num* is returned by a previously executed FOPEN function. If *file_num* is not valid, FWRITELN returns the #VALUE! error value.

***Text*** is the text you want to write to the file.

***Example***

See the first example in FPOS.

***Related Functions***

| | |
|---|---|
| FOPEN | Opens a text file |
| FREAD | Reads characters from a text file |
| FWRITE | Writes characters to a text file |

*Macro Sheets Only*

## GALLERY.3D.AREA(type_num)
## GALLERY.3D.AREA?(type_num)

Equivalent to the Gallery 3-D Area command. Changes the active chart to a 3-D area chart.

***Type_num*** is the number of a format in the Gallery 3-D Area dialog box that you want to apply to the 3-D area chart.

***Example***

The following macro formula applies the first format in the Gallery 3-D Area dialog box to the main chart:

```
GALLERY.3D.AREA(1)
```

*Macro Sheets Only*

## GALLERY.3D.COLUMN(type_num)
## GALLERY.3D.COLUMN?(type_num)

Equivalent to the Gallery 3-D Column command. Changes the active chart to a 3-D column chart.

***Type_num*** is the number of a format in the Gallery 3-D Column dialog box that you want to apply to the 3-D column chart.

***Example***

The following macro formula applies the first format in the Gallery 3-D Column dialog box to the main chart:

```
GALLERY.3D.COLUMN(1)
```

*Macro Sheets Only*

## GALLERY.3D.LINE(type_num)
## GALLERY.3D.LINE?(type_num)

Equivalent to the Gallery 3-D Line command. Changes the active chart to a 3-D line chart.

***Type_num*** is the number of a format in the Gallery 3-D Line dialog box that you want to apply to the 3-D line chart.

***Example***

The following macro formula applies the first format in the Gallery 3-D Line dialog box to the main chart:

```
GALLERY.3D.LINE(1)
```

*Macro Sheets Only*

## GALLERY.3D.PIE(type_num)
## GALLERY.3D.PIE?(type_num)

Equivalent to the Gallery 3-D Pie command. Changes the active chart to a 3-D pie chart.

***Type_num*** is the number of a format in the Gallery 3-D Pie dialog box that you want to apply to the 3-D pie chart.

***Example***

The following macro formula applies the first format in the Gallery 3-D Pie dialog box to the main chart:

```
GALLERY.3D.PIE(1)
```

*Macro Sheets Only*

## GALLERY.AREA(type_num,delete_overlay)
## GALLERY.AREA?(type_num,delete_overlay)

Equivalent to the Gallery Area command. Changes the format of the active chart to an area chart. For detailed information on the Gallery Area command, see online Help.



***Type_num*** is the number of a format in the Gallery Area dialog box that you want to apply to the area chart.

***Delete_overlay*** is a logical value specifying whether to delete an overlay chart.

- If *delete_overlay* is TRUE, Microsoft Excel deletes the overlay chart, if present, and applies the new format to the main chart.
- If *delete_overlay* is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay chart, depending on the location of the selected series.

***Example***

The following macro formula deletes the overlay chart, if it is present, and applies the first format in the gallery to the main chart:

```
GALLERY.AREA(1,TRUE)
```

*Macro Sheets Only*

## GALLERY.BAR(type_num,delete_overlay)
## GALLERY.BAR?(type_num,delete_overlay)

Equivalent to the Gallery Bar command. Changes the format of the active chart to a bar chart. For detailed information on the Gallery Bar command, see online Help.

***Type_num*** is the number of a format in the Gallery Bar dialog box that you want to apply to the bar chart.

***Delete_overlay*** is a logical value specifying whether to delete an overlay chart.

- If *delete_overlay* is TRUE, Microsoft Excel deletes the overlay chart, if present, and applies the new format to the main chart.
- If *delete_overlay* is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay chart, depending on the location of the selected series.

*Macro Sheets Only*

## GALLERY.COLUMN(type_num,delete_overlay)
## GALLERY.COLUMN?(type_num, delete_overlay)

Equivalent to the Gallery Column command. Changes the format of the active chart to a column chart. For detailed information on the Gallery Column command, see online Help.

***Type_num*** is the number of a format in the Gallery Column dialog box that you want to apply to the column chart.

***Delete_overlay*** is a logical value specifying whether to delete an overlay chart.

- If *delete_overlay* is TRUE, Microsoft Excel deletes the overlay chart, if present, and applies the new format to the main chart.
- If *delete_overlay* is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay chart, depending on the location of the selected series.

***Example***

The following macro formula deletes the overlay chart, if it is present, and applies the first format in the gallery to the main chart:

```
GALLERY.COLUMN(1,TRUE)
```

*Macro Sheets Only*

## GALLERY.LINE(type_num,delete_overlay)
## GALLERY.LINE?(type_num,delete_overlay)

Equivalent to the Gallery Line command. Changes the format of the active chart to a line chart. For detailed information on the Gallery Line command, see online Help.

***Type_num*** is the number of a format in the Gallery Line dialog box that you want to apply to the line chart.

***Delete_overlay*** is a logical value specifying whether to delete an overlay chart.

- If *delete_overlay* is TRUE, Microsoft Excel deletes the overlay chart, if present, and applies the new format to the main chart.
- If *delete_overlay* is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay chart, depending on the location of the selected series.

### Example

The following macro formula deletes the overlay chart, if it is present, and applies the first format in the gallery to the main chart:

```
GALLERY.LINE(1,TRUE)
```

*Macro Sheets Only*

## GALLERY.PIE(type_num,delete_overlay)
## GALLERY.PIE?(type_num,delete_overlay)

Equivalent to the Gallery Pie command. Changes the format of the active chart to a pie chart. For detailed information on the Gallery Pie command, see online Help.

***Type_num*** is the number of a format in the Gallery Pie dialog box that you want to apply to the pie chart.

***Delete_overlay*** is a logical value specifying whether to delete an overlay chart.

- If *delete_overlay* is TRUE, Microsoft Excel deletes the overlay chart, if present, and applies the new format to the main chart.
- If *delete_overlay* is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay chart, depending on the location of the selected series.

### Example

The following macro formula deletes the overlay chart, if it is present, and applies the first format in the gallery to the main chart:

```
GALLERY.PIE(1,TRUE)
```

*Macro Sheets Only*

## GALLERY.SCATTER(type_num, delete_overlay)
## GALLERY.SCATTER?(type_num, delete_overlay)

Equivalent to the Gallery Scatter command. Changes the format of the active chart to an xy (scatter) chart. For detailed information on the Gallery Scatter command, see online Help.

***Type_num*** is the number of a format in the Gallery Scatter dialog box that you want to apply to the xy (scatter) chart.

***Delete_overlay*** is a logical value specifying whether to delete an overlay chart.

- If *delete_overlay* is TRUE, Microsoft Excel deletes the overlay chart, if present, and applies the new format to the main chart.
- If *delete_overlay* is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay chart, depending on the location of the selected series.

### Example

The following macro formula deletes the overlay chart, if it is present, and applies the first format in the gallery to the main chart:

```
GALLERY.SCATTER(1,TRUE)
```

*Macro Sheets Only*

## GET.BAR()

**Form 1: Menu bar number**

Returns the number of the active menu bar. There are two forms of GET.BAR. Use this form in macros when you need to know which menu bar is active so you can refer to it in other functions that manipulate menu bars. For a list of the ID numbers for Microsoft Excel's built-in menu bars, see ADD.COMMAND.

For information on menu bars, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.



### Example

The following macro formula adds the command named ReturnMenu to the File menu when the worksheet and macro sheet menu bar (full menus) is displayed:

```
IF(GET.BAR()=1,ADD.COMMAND(1,"File",ReturnMenu))
```

### Related Functions

| | |
|---|---|
| ADD.BAR | Creates a new menu bar |
| DELETE.BAR | Deletes a menu bar |
| SHOW.BAR | Displays a menu bar |

*Macro Sheets Only*

## GET.BAR(bar_num,menu,command)

**Form 2: Command name or number**

Returns the name or position number of a specified command on a menu. There are two forms of GET.BAR. Use this form to return information that you can use with functions that add, delete, or alter menu commands.

***Bar_num*** is the number of a menu bar in which the command you want information about resides. *Bar_num* can be the number of a built-in menu bar or the number returned by a previously run ADD.BAR function. For a list of the ID numbers for Microsoft Excel's built-in menu bars, see ADD.COMMAND.

***Menu*** is the menu on which the command resides. *Menu* can be the name of the menu as text or the number of the menu. Menus are numbered starting with 1 from the left of the screen.

***Command*** is the command whose name or number you want returned. *Command* can be the name of the command as text, in which case the number is returned, or the number of the command on the menu, in which case the name is returned. The top command on a menu is in position 1. If *command* is 0, the name or position number of the menu is returned.

### Notes

- If an ampersand is used to indicate the access key in the name of a custom command, the ampersand is included in the name returned by GET.BAR. All built-in commands have an ampersand before the letter used as the access key.
- If the command name or position specified does not exist, GET.BAR returns the #N/A error value.

### Examples

In the default worksheet and macro sheet menu bar (full menus):

```
GET.BAR(1,"File","Print...") equals 13
GET.BAR(1,"File",13) equals "&Print..."
```

### Related Functions

| | |
|---|---|
| ADD.COMMAND | Adds a command to a menu |
| DELETE.COMMAND | Deletes a command from a menu |
| RENAME.COMMAND | Renames a command |

*Macro Sheets Only*

## GET.CELL(type_num,reference)

Returns information about the formatting, location, or contents of the cell in the upper-left corner of the first area in a reference or the current selection. Use GET.CELL to get information about a cell to use in a macro that behaves according to the status of a cell.

***Type_num*** is a number that specifies what type of cell information you want. The following list shows the possible values of *type_num* and the corresponding results:

| *Type_num* | Returns |
|---|---|
| 1 | Absolute reference of the upper-left cell in *reference*, as text in the current workspace reference style. |
| 2 | Row number of the top cell in *reference*. |

| *Type_num* | Returns |
|---|---|
| 3 | Column number of the leftmost cell in *reference*. |
| 4 | TYPE(*reference*). |
| 5 | Contents of *reference*. |
| 6 | Formula in *reference*, as text, in either A1 or R1C1 format depending on the workspace setting. |
| 7 | Number format of the cell, as text (for example, "m/d/yy" or "General"). |
| 8 | Number indicating the cell's alignment:<br>1 = General<br>2 = Left<br>3 = Center<br>4 = Right<br>5 = Fill |
| 9 | Number indicating the left-border style assigned to the cell:<br>0 = No border<br>1 = Thin line<br>2 = Medium line<br>3 = Dashed line<br>4 = Dotted line<br>5 = Thick line<br>6 = Double line<br>7 = Hairline |
| 10 | Number indicating the right-border style assigned to the cell. See *type_num* 9 for descriptions of the numbers returned. |
| 11 | Number indicating the top-border style assigned to the cell. See *type_num* 9 for descriptions of the numbers returned. |
| 12 | Number indicating the bottom-border style assigned to the cell. See *type_num* 9 for descriptions of the numbers returned. |
| 13 | Number from 0 to 18, indicating the pattern of the selected cell as displayed in the Format Patterns dialog box. If no pattern selected, returns 0. |
| 14 | If cell is locked, returns TRUE; otherwise FALSE. |
| 15 | If cell is hidden, returns TRUE; otherwise FALSE. |
| 16 | Column width of cell, measured in characters of the Normal font for the document. |
| 17 | Row height of cell, in points. |
| 18 | Name of font, as text. |
| 19 | Size of font, in points. |
| 20 | If cell is bold, returns TRUE; otherwise FALSE. |
| 21 | If cell is italic, returns TRUE; otherwise FALSE. |
| 22 | If cell is underlined, returns TRUE; otherwise FALSE. |
| 23 | If cell is struck through, returns TRUE; otherwise FALSE. |
| 24 | Font color as a number in the range 1 to 16. If font color is automatic, returns 0. |
| 25 | If cell is outlined, returns TRUE; otherwise FALSE. Outline font format is not supported by Microsoft Excel for Windows or OS/2. |
| 26 | If cell is shadowed, returns TRUE; otherwise FALSE. Shadow font format is not supported by Microsoft Excel for Windows or OS/2. |
| 27 | Number indicating whether a page break occurs at the cell:<br>0 = No break<br>1 = Row<br>2 = Column<br>3 = Both row and column |
| 28 | Row level (outline). |



| *Type_num* | Returns |
|---|---|
| 29 | Column level (outline). |
| 30 | If the row containing the active cell is a summary row, returns TRUE; otherwise FALSE. |
| 31 | If the column containing the active cell is a summary column, returns TRUE; otherwise FALSE. |
| 32 | Name of the sheet containing the cell. |
| 33 | If cell is formatted to wrap, returns TRUE; otherwise FALSE. |
| 34 | Left-border color as a number in the range 1 to 16. If color is automatic, returns 0. |
| 35 | Right-border color as a number in the range 1 to 16. If color is automatic, returns 0. |
| 36 | Top-border color as a number in the range 1 to 16. If color is automatic, returns 0. |
| 37 | Bottom-border color as a number in the range 1 to 16. If color is automatic, returns 0. |
| 38 | Shade foreground color as a number in the range 1 to 16. If color is automatic, returns 0. |
| 39 | Shade background color as a number in the range 1 to 16. If color is automatic, returns 0. |
| 40 | Style of the cell, as text. |

***Reference*** is a cell or a range of cells from which you want information.

- If *reference* is a range of cells, the cell in the upper-left corner of the first range in *reference* is used.
- If *reference* is omitted, the active cell is assumed.

### *Examples*

The following macro formula returns TRUE if cell B4 on worksheet Sheet1 is bold:

```
GET.CELL(20,Sheet1!$B$4)
```

You can use the information returned by GET.CELL to initiate an action. The following macro formula runs a custom function named BoldCell if the GET.CELL formula returns FALSE:

```
IF(GET.CELL(20,Sheet1!$B$4),,BoldCell())
```

### *Related Functions*

| | |
|---|---|
| ABSREF | Returns the absolute reference of a cell |
| ACTIVE.CELL | Returns the reference of the active cell |
| GET.FORMULA | Returns the contents of a cell |
| GET.NAME | Returns the definition of a name |
| GET.NOTE | Returns characters from a note |
| RELREF | Returns the relative reference of a cell |

*Macro Sheets Only*

## GET.CHART.ITEM(*x_y_index,point_index, item_text*)

Returns the vertical or horizontal position of a point on a chart item. Use these position numbers with FORMAT.MOVE and FORMAT.SIZE to change the position and size of chart items.

***X_y_index*** is a number specifying which of the coordinates you want returned.

| *X_y_index* | Coordinate returned |
|---|---|
| 1 | Horizontal coordinate |
| 2 | Vertical coordinate |

***Point_index*** is a number specifying the point on the chart item. These indexes are described later. If *point_index* is omitted, it is assumed to be 1.

If the specified item is a point, *point_index* must be 1.

If the specified item is any line other than a data line, use these values for *point_index*:

| Point_Index | Chart item position |
|---|---|
| 1 | Lower or left |
| 2 | Upper or right |

If the selected item is a rectangle or an area in an area chart, use these values for *point_index*:

| Point_Index | Chart item position |
|---|---|
| 1 | Upper left |
| 2 | Upper middle |
| 3 | Upper right |
| 4 | Right middle |
| 5 | Lower right |
| 6 | Lower middle |
| 7 | Lower left |
| 8 | Left middle |

If the selected item is an arrow, use these values for *point_index*:

| Point_Index | Chart item position |
|---|---|
| 1 | Arrow shaft |
| 2 | Arrowhead |

If the selected item is a pie slice, use these values for *point_index*:

| Point_Index | Chart item position |
|---|---|
| 1 | Outermost counterclockwise point |
| 2 | Outer center point |
| 3 | Outermost clockwise point |
| 4 | Midpoint of the most clockwise radius |
| 5 | Center point |
| 6 | Midpoint of the most counterclockwise radius |

### Note

If the specified item does not exist, or if a chart is not active when the function is carried out, the #VALUE! error value is returned.

***Item_text*** is a selection code that specifies which item of a chart to select. See the chart form of SELECT for the *item_text* codes to use for each item of a chart.

- If *item_text* is omitted, it is assumed to be the currently selected item.
- If *item_text* is omitted and no item is selected, GET.CHART.ITEM returns the #VALUE! error value.

### Example

The following macro formulas return the horizontal and vertical locations, respectively, of the top of the main-chart value axis:

```
GET.CHART.ITEM(1,2,"Axis 1")
GET.CHART.ITEM(2,2,"Axis 1")
```

You could then use FORMAT.MOVE to move a floating text item to the position returned by these two formulas.

### Related Functions

| | |
|---|---|
| GET.DOCUMENT | Returns information about a document |
| GET.FORMULA | Returns the contents of a cell |

*Macro Sheets Only*

## GET.DEF(*def_text,document_text,type_num*)

Returns the name, as text, that is defined for a particular area, value, or formula in a document. Use GET.DEF to get the name corresponding to a definition. To get the definition of a name, use GET.NAME.

***Def_text*** can be anything you can define a name to refer to, including a reference, a value, or a formula.

- References must be given in R1C1 style, such as "R3C5".



- If *def_text* is a value or formula, it is not necessary to include the equal sign that is displayed in the Refers To box in the Define Name dialog box.
- If there is more than one name for *def_text*, GET.DEF returns the first name. If no name matches *def_text*, GET.DEF returns the #NAME? error value.

***Document_text*** specifies the worksheet or macro sheet that *def_text* is on. If *document_text* is omitted, it is assumed to be the active macro sheet.

***Type_num*** is a number from 1 to 3 specifying which types of names are returned.

| Type_num | Type of name returned |
|---|---|
| 1 or omitted | Normal names only |
| 2 | Hidden names only |
| 3 | All names |

### Examples

If the specified range in Sheet4 is named Sales, the following macro formula returns "Sales":

```
GET.DEF("R2C2:R9C6","Sheet4")
```

If the value 100 in Sheet4 is defined as Constant, the following macro formula returns "Constant":

```
GET.DEF("100","Sheet4")
```

If the specified formula in Sheet4 is named SumTotal, the following macro formula returns "SumTotal":

```
GET.DEF("SUM(R1C1:R10C1)","Sheet4")
```

If 3 is defined as the hidden name Counter on the active macro sheet, the following macro formula returns "Counter":

```
GET.DEF("3",,2)
```

### Related Functions

| | |
|---|---|
| GET.CELL | Returns information about the formatting, location, or contents of a cell |
| GET.NAME | Returns the definition of a name |
| GET.NOTE | Returns characters from a note |
| NAMES | Returns a horizontal text array of all names defined on a document |

*Macro Sheets Only*

## GET.DOCUMENT(*type_num,name_text*)

Returns information about a document.

***Type_num*** is a number that specifies what type of document information you want. The following lists show the possible values of *type_num* and the corresponding results.

| Type_num | Returns |
|---|---|
| 1 | Name of the document *name_text*, as text. The document name does not include the drive, directory or folder, or window number. |
| 2 | Pathname of the directory or folder containing *name_text*, as text. If the document *name_text* hasn't been saved yet, returns #N/A. |
| 3 | Number indicating the type of document:<br>1 = Worksheet<br>2 = Chart<br>3 = Macro sheet<br>4 = Info window is active |
| 4 | If changes have been made to the document since it was last saved, returns TRUE; otherwise FALSE. |
| 5 | If document is read-only, returns TRUE; otherwise FALSE. |
| 6 | If file is protected, returns TRUE; otherwise FALSE. |
| 7 | If document cells are protected, returns TRUE; otherwise FALSE. |
| 8 | If document windows are protected, returns TRUE; otherwise FALSE. |

These four values of *type_num* apply only to charts.

| *Type_num* | Returns |
|---|---|
| 9 | Number indicating the type of the main chart:<br>1 = 2-D area<br>2 = 2-D bar<br>3 = 2-D column<br>4 = 2-D line<br>5 = 2-D pie<br>6 = 2-D xy (scatter)<br>7 = 3-D area<br>8 = 3-D column<br>9 = 3-D line<br>10 = 3-D pie |
| 10 | Number indicating the type of the overlay chart. Same numbers as for main chart, above. If there is no overlay chart, returns #N/A. |
| 11 | Number of series in the main chart. |
| 12 | Number of series in the overlay chart. |

These values of *type_num* apply only to worksheets and macro sheets, except for *type_num* 34, 35, 36, 37, and 40, which also apply to charts.

| *Type_num* | Returns |
|---|---|
| 9 | Number of the first used row. If document is empty, returns 0. |
| 10 | Number of the last used row. If document is empty, returns 0. |
| 11 | Number of the first used column. If document is empty, returns 0. |
| 12 | Number of the last used column. If document is empty, returns 0. |
| 13 | Number of windows. |
| 14 | Number indicating calculation mode:<br>1 = Automatic<br>2 = Automatic Except Tables<br>3 = Manual |
| 15 | If iteration is enabled, returns TRUE; otherwise FALSE. |
| 16 | Maximum number of iterations. |
| 17 | Maximum change between iterations. |
| 18 | If updating remote references is enabled, returns TRUE; otherwise FALSE. |
| 19 | If set to Precision As Displayed in the Options Calculation dialog box, returns TRUE; otherwise FALSE. |
| 20 | If document is set to 1904 Date System in the Options Calculation dialog box, returns TRUE; otherwise FALSE. |
| 21 | Four-item horizontal text array of the names of the four default fonts. |
| 22 | Four-item horizontal number array of the sizes of the four default fonts. |
| 23 | Four-item horizontal logical array indicating which of the four default fonts are bold. (For example, if fonts 1 and 4 are bold, returns the array {TRUE,FALSE,FALSE, TRUE}.) |
| 24 | Four-item horizontal logical array indicating which of the four default fonts are italic. |
| 25 | Four-item horizontal logical array indicating which of the four default fonts are underlined. |
| 26 | Four-item horizontal logical array indicating which of the four default fonts are struck through. |
| 27 | Four-item horizontal array indicating the color of the four default fonts where color is a number from 1 to 16, and 0 is automatic. |
| 28 | Four-item horizontal array indicating which of the four default fonts are outlined. Always returns FALSE in Microsoft Excel for Windows and OS/2. Included for compatibility with Microsoft Excel for the Macintosh. |



| *Type_num* | Returns |
|---|---|
| 29 | Four-item horizontal array indicating which of the four default fonts are shadowed. Always returns FALSE in Microsoft Excel for Windows and OS/2. Included for compatibility with Microsoft Excel for the Macintosh. |
| 30 | Horizontal array of consolidation references for the current document, in the form of text. If the list is empty, returns #NA. |
| 31 | Number from 1 to 11, indicating the function used in the current consolidation. The function that corresponds to each number is listed under the CONSOLIDATE function earlier in this book. If no function is specified, returns 9. |
| 32 | Three-item horizontal array indicating the status of the check boxes in the Data Consolidate dialog box. An item is TRUE if the check box is turned on or FALSE if the check box is turned off. The first cell indicates the Top Row check box, the second cell indicates the Left Column check box, and the third cell indicates the Create Links to Source Data check box. |
| 33 | If Recalculate Before Saving is turned on in the Options Calculation dialog box, returns TRUE; otherwise FALSE. |
| 34 | If the document is read-only recommended, returns TRUE; otherwise FALSE. |
| 35 | If the document is write reserved, returns TRUE; otherwise FALSE. |
| 36 | Name of the user with current write permission for the document. |
| 37 | Number corresponding to the file type of the document as displayed in the File Save As dialog box. See SAVE.AS for a list of the file types. |
| 38 | If Summary Rows Below Detail is turned on in the Formula Outline dialog box, returns TRUE; otherwise FALSE. |
| 39 | If Summary Columns To Right Of Detail is turned on in the Formula Outline dialog box, returns TRUE; otherwise FALSE. |
| 40 | If Create Backup File is turned on in the File Save As dialog box, returns TRUE; otherwise FALSE. |
| 41 | Number from 1 to 3 indicating whether objects are displayed:<br>1 = All objects are displayed<br>2 = Placeholders for pictures and charts<br>3 = All objects are hidden |
| 42 | Horizontal array of all objects in the sheet. If there are no objects, returns #N/A. |
| 43 | If Save External Link Values is turned on in the Options Calculation dialog box, returns TRUE; otherwise FALSE. |
| 44 | If document objects are protected, returns TRUE; otherwise FALSE. |

***Name_text*** is the name of an open document. If *name_text* is omitted, it is assumed to be the active document.

### Examples

The following macro formula returns TRUE if the contents of the active document are protected:

```
GET.DOCUMENT(7)
```

To find out if Sheet1 is password-protected and if its contents and windows are protected, enter the following formula in a three-cell horizontal array:

```
GET.DOCUMENT({6,7,8},"Sheet1")
```

In Microsoft Excel for Windows or OS/2, the following macro formula returns the number of windows in SALES.XLS:

```
GET.DOCUMENT(13,"SALES.XLS")
```

In Microsoft Excel for the Macintosh, the following macro formula returns 3 if the overlay on Sales Chart is a column:

GET.DOCUMENT(10,"Sales Chart")

### Related Functions

| | |
|---|---|
| GET.CELL | Returns information about the formatting, location, or contents of a cell |
| GET.WINDOW | Returns information about a window |
| GET.WORKSPACE | Returns information about the workspace |

Macro Sheets Only

## GET.FORMULA (reference)

Returns the contents of a cell as they would appear in the formula bar. The contents are given as text, for example, "=2*PI()/360". If the formula contains references, they are returned as R1C1-style references, such as "=RC[1]*(1+R1C1)". Use GET.FORMULA to get a formula from a cell in order to edit its arguments. Use GET.CELL(6) to get a formula in either A1 or R1C1 format, depending on the workspace setting.

**Reference** is a cell or range of cells on a worksheet or macro sheet.

- If a range of cells is selected, GET.FORMULA returns the contents of the upper-left cell in *reference*.
- *Reference* can be an external reference.

### Tip

If you want to get the formula in the active cell, use the ACTIVE.CELL function as the *reference* argument.

### Examples

If cell A3 on the active worksheet contains the number 523, then:

GET.FORMULA(!$A$3) equals "523"

If cell C2 on the active worksheet contains the formula =B2*(1+$A$1), then:

GET.FORMULA(!$C$2) equals "=RC[-1]*(1+R1C1)"

The following formula returns the contents of the active cell on the active document:

GET.FORMULA(ACTIVE.CELL())

### Related Functions

| | |
|---|---|
| GET.CELL | Returns information about the formatting, location, or contents of a cell |
| GET.DEF | Returns a name matching a definition |
| GET.NAME | Returns the definition of a name |
| GET.NOTE | Returns characters from a note |

Macro Sheets Only

## GET.LINK.INFO(link_text,type_num,type_of_link, reference)

Returns information about the specified link. Use GET.LINK.INFO to get information about the update settings of a link.

**Link_text** is the pathname of the link as displayed in the File Links dialog box.

**Type_num** is a number that specifies what type of information about the currently selected link to return. *Type_num* 2 applies only to publishers and subscribers in Microsoft Excel for the Macintosh.

| Type_num | Returns |
|---|---|
| 1 | If the link is set to automatic update, returns 1; otherwise 2. |
| 2 | Date of the latest edition as a serial number. Returns #N/A if *link_text* is not a publisher or a subscriber. |

**Type_of_link** is a number from 1 to 6 that specifies what type of link you want to get information about.



| Type_of_link | Link document type |
|---|---|
| 1 | Not applicable |
| 2 | DDE link |
| 3 | Not applicable |
| 4 | Outgoing NewWave link |
| 5 | Publisher |
| 6 | Subscriber |

**Reference** specifies the cell range in R1C1 format of the publisher or subscriber that you want information about. *Reference* is required if you have more than one publisher or subscriber of a single edition name on the active document. Use *reference* to specify the location of the subscriber you want to return information about. If the subscriber is a picture, or if the publisher is an embedded chart, *reference* is the number of the object as displayed in the reference area.

### Notes

- If Microsoft Excel cannot find *link_text*, or if *type_of_link* does not match the link specified by *link_text*, GET.LINK.INFO returns the #VALUE! error value.
- If you have more than one subscriber to the edition *link_text* or if the same area is published more than once, you must specify *reference*.

### Related Functions

| | |
|---|---|
| CREATE.PUBLISHER | Creates a publisher in a Microsoft Excel document |
| SUBSCRIBE.TO | Inserts contents of an edition into the active document |
| UPDATE.LINK | Updates a link to another document |

Macro Sheets Only

## GET.NAME(name_text)

Returns the definition of a name as it appears in the Refers To box of the Formula Define Name dialog box. If the definition contains references, they are given as R1C1-style references. Use GET.NAME to check the value defined by a name. To get the name corresponding to a definition, use GET.DEF.

**Name_text** can be a name defined on the macro sheet; an external reference to a name defined on the active document, for example, "!Sales"; or an external reference to a name defined on a particular worksheet, for example, "Sheet1!Sales". *Name_text* can also be a hidden name.

### Note

If the Cells check box has been turned on in the Options Protect Document dialog box to protect the document containing the name, GET.NAME returns the #N/A error value.

### Examples

If the name Sales on a macro sheet is defined as the number 523, then:

GET.NAME("Sales") equals "523"

If the name Profit on the active worksheet is defined as the formula =Sales–Costs, then:

GET.NAME("!Profit") equals "=Sales–Costs"

If the name Database on the active worksheet is defined as the range A1:F500, then:

GET.NAME("!Database") equals "=R1C1:R500C6"

### Related Functions

| | |
|---|---|
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| GET.CELL | Returns information about the formatting, location, or contents of a cell |

| | |
|---|---|
| GET.DEF | Returns a name matching a definition |
| NAMES | Returns a horizontal text array of all the names defined on a document |
| SET.NAME | Defines a name on a macro sheet |

*Macro Sheets Only*

## GET.NOTE(*cell_ref,start_char,num_chars*)

Returns characters from a note. Use GET.NOTE to move the contents of a note to a cell or text box or to another cell note.

***Cell_ref*** is the cell to which the note is attached. If *cell_ref* is omitted, the note attached to the active cell is returned.

***Start_char*** is the number of the first character in the note to return. If *start_char* is omitted, it is assumed to be 1, the first character in the note.

***Num_chars*** is the number of characters to return. *Num_chars* must be less than or equal to 255. If *num_chars* is omitted, it is assumed to be the length of the note attached to *cell_ref*.

### Examples

The following macro formula returns the first 200 characters in the note attached to cell A3 on the active sheet:

```
GET.NOTE(!$A$3,1,200)
```

In Microsoft Excel for Windows or OS/2, the following macro formula returns the 10th through the 39th characters of the note attached to cell C2 on SALES.XLS.

```
GET.NOTE("SALES.XLS!R2C3",10,30)
```

In Microsoft Excel for the Macintosh, the following macro formula returns the 10th through the 39th characters of the note attached to cell C2 on Sales:

```
GET.NOTE("Sales!R2C3",10,30)
```

### Related Functions

| | |
|---|---|
| GET.CELL | Returns information about a cell |
| NOTE | Creates or changes a cell note |

*Macro Sheets Only*

## GET.OBJECT(*type_num,object_id_text,start_num, count_num*)

Returns information about the specified object. Use GET.OBJECT to return information you can use in other macro formulas that manipulate objects.

***Type_num*** is a number specifying the type of information you want returned about an object. GET.OBJECT returns the #VALUE! error value if an object isn't specified or if more than one object is selected.

| *Type_num* | Returns |
|---|---|
| 1 | Number specifying the type of the selected object:<br>1 = Line<br>2 = Rectangle<br>3 = Oval<br>4 = Arc<br>5 = Embedded chart<br>6 = Text box<br>7 = Button<br>8 = Picture |
| 2 | If the object is locked, returns TRUE; otherwise FALSE. |
| 3 | Z-order position (layering) of the object. |
| 4 | Reference of the cell under the upper-left corner of the object as text in R1C1 reference style; for a line or arc, returns the start point. |
| 5 | X offset from the upper left corner of the cell under the upper-left corner of the object, measured in points. |
| 6 | Y offset from the upper-left corner of the cell under the upper-left corner of the object, measured in points. |
| 7 | Reference of the cell under the lower-right corner of the object as text in R1C1 reference style; for a line or arc, returns the end point. |



| *Type_num* | Returns |
|---|---|
| 8 | X offset from the upper-left corner of the cell under the lower-right corner of the object, measured in points. |
| 9 | Y offset from the upper-left corner of the cell under the lower-right corner of the object, measured in points. |
| 10 | Name, including the filename, of the macro assigned to the object. If no macro is assigned, returns FALSE. |
| 11 | Number indicating how the object moves and sizes:<br>1 = Object moves and sizes with cells<br>2 = Object moves with cells<br>3 = Object is fixed |

Values 12 to 21 for *type_num* apply only to text boxes and buttons. If another type of object is selected, GET.OBJECT returns the #VALUE! error value.

| *Type_num* | Returns |
|---|---|
| 12 | Text starting at *start_num* for *count_num* characters. |
| 13 | Font name of all text starting at *start_num* for *count_num* characters. If the text contains more than one font name, returns #N/A. |
| 14 | Font size of all text starting at *start_num* for *count_num* characters. If the text contains more than one font size, returns #N/A. |
| 15 | If all text starting at *start_num* for *count_num* characters is bold, returns TRUE. If text contains only partial bold formatting, returns #N/A. |
| 16 | If all text starting at *start_num* for *count_num* characters is italic, returns TRUE. If text contains only partial italic formatting, returns #N/A. |
| 17 | If all text starting at *start_num* for *count_num* characters is underlined, returns TRUE. If text contains only partial underline formatting, returns #N/A. |
| 18 | If all text starting at *start_num* for *count_num* characters is struck through, returns TRUE. If text contains only partial struck through formatting, returns #N/A. |
| 19 | If all text starting at *start_num* for *count_num* characters is outlined, returns TRUE. If text contains only partial outline formatting, returns #N/A. Always returns FALSE in Microsoft Excel for Windows or OS/2. |
| 20 | If all text starting at *start_num* for *count_num* characters is shadowed, returns TRUE. If text contains only partial shadow formatting, returns #N/A. Always returns FALSE in Microsoft Excel for Windows or OS/2. |
| 21 | Number from 0 to 16 indicating the color of all text starting at *start_num* for *count_num* characters; if color is automatic, returns 0. If more than one color is used, returns #N/A. |

Values 22 to 25 for *type_num* also apply only to text boxes and buttons. If another type of object is selected, GET.OBJECT returns the #N/A error value.

| *Type_num* | Returns |
|---|---|
| 22 | Number indicating the horizontal alignment of text:<br>1 = Left<br>2 = Center<br>3 = Right |
| 23 | Number indicating the vertical alignment of text:<br>1 = Top<br>2 = Center<br>3 = Bottom |
| 24 | Number indicating the orientation of text:<br>0 = Horizontal<br>1 = Vertical<br>2 = Upward<br>3 = Downward |

| *Type_num* | Returns |
|---|---|
| 25 | If button or text box is set to automatic sizing, returns TRUE; otherwise FALSE. |

The following values for *type_num* apply to all objects, except where indicated.

| *Type_num* | Returns |
|---|---|
| 26 | If object is visible, returns TRUE; if object has been hidden by the HIDE.OBJECT function, returns FALSE. |
| 27 | Number indicating the type of the border or line:<br>0 = Custom<br>1 = Automatic<br>2 = None |
| 28 | Number indicating the style of the border or line as shown in the Style box in the Format Patterns dialog box:<br>0 = None<br>1 = Solid line<br>2 = Dashed line<br>3 = Dotted line<br>4 = Dashed dotted line<br>5 = Dash double-dotted line<br>6 = 50% gray line<br>7 = 75% gray line<br>8 = 25% gray line |
| 29 | Number from 0 to 16 indicating the color of the border or line; if border is automatic, returns 0. |
| 30 | Number indicating the weight of the border or line:<br>0 = Hairline<br>1 = Thin<br>2 = Medium<br>3 = Thick |
| 31 | Number indicating the type of fill:<br>0 = Custom<br>1 = Automatic<br>2 = None |
| 32 | Number from 1 to 18 indicating the fill pattern as shown in the Format Patterns dialog box. |
| 33 | Number from 0 to 16 indicating the foreground color of the fill pattern; if fill is automatic, returns 0. If the object is a line, returns #N/A. |
| 34 | Number from 0 to 16 indicating the background color of the fill pattern; if fill is automatic, returns 0. If the object is a line, returns #N/A. |
| 35 | Number indicating the width of the arrowhead:<br>1 = Narrow<br>2 = Medium<br>3 = Wide<br>If the object is not a line, returns #N/A. |
| 36 | Number indicating the length of the arrowhead:<br>1 = Short<br>2 = Medium<br>3 = Long<br>If the object is not a line, returns #N/A. |
| 37 | Number indicating the style of the arrowhead:<br>1 = No head<br>2 = Open head<br>3 = Closed head<br>If the object is not a line, returns #N/A. |
| 38 | If the border has round corners, returns TRUE; if the corners are square, returns FALSE. If the object is a line, returns #N/A. |
| 39 | If the border has a shadow, returns TRUE; if the border has no shadow, returns FALSE. If the object is a line, returns #N/A. |
| 40 | If the Lock Text check box in the Format Object Protection dialog box is turned on, returns TRUE; otherwise FALSE. |

***Object_id_text*** is the name and number, or number alone, of the object you want information about. *Object_id_text* is the text displayed in the reference area when the object is selected. If *object_id_text* is omitted, it is assumed to be the selected object. If *object_id_text* is omitted and no object is selected, GET.OBJECT returns the #REF! error value and interrupts the macro.



***Start_num*** is the number of the first character in the text box you want information about. *Start_num* is ignored unless a text box or button is specified by *type_num* and *object_id_text*. If *start_num* is omitted, it is assumed to be 1.

***Count_num*** is the number of characters in a text box, starting at *start_num*, that you want information about. *Count_num* is ignored unless a text box or button is specified by *type_num* and *object_id_text*. If *count_num* is omitted, it is assumed to be 255.

### *Examples*

The following macro formula returns the reference of the cell under the upper-left corner of the object Oval 3:

```
GET.OBJECT(4,"Oval 3")
```

The following macro formula changes the protection status of the object Rectangle 2 if it is locked:

```
IF(GET.OBJECT(2,"Rectangle 2"),
OBJECT.PROTECTION(FALSE))
```

The following macro formula returns characters 25 through 185 from the object Text 5:

```
GET.OBJECT(12,"Text 5",25,185)
```

### *Related Functions*

| | |
|---|---|
| CREATE.OBJECT | Creates the specified object |
| FORMAT.FONT | Applies a font to the current selection |
| OBJECT.PROTECTION | Controls how an object is protected |
| PLACEMENT | Determines an object's relationship to underlying cells |

*Macro Sheets Only*

## *GET.WINDOW(type_num,window_text)*

Returns information about a window. Use GET.WINDOW in a macro that depends on the status of a window, such as its name, size, position, and display options.

***Type num*** is a number that specifies what type of window information you want. The following list shows the possible values of *type_num* and the corresponding results:

| *Type_num* | Returns |
|---|---|
| 1 | Name of the document in the window as text |
| 2 | Number of the window |
| 3 | X position, measured in points from the left edge of the workspace (in Microsoft Excel for Windows or OS/2) or screen (in Microsoft Excel for the Macintosh) to the left edge of the window |
| 4 | Y position, measured in points from the top edge of the workspace (in Microsoft Excel for Windows or OS/2) or screen (in Microsoft Excel for the Macintosh) to the top edge of the window |
| 5 | Width, measured in points |
| 6 | Height, measured in points |
| 7 | If window is hidden, returns TRUE; otherwise FALSE |

The rest of the values for *type_num* apply only to worksheets and macro sheets:

| *Type_num* | Returns |
|---|---|
| 8 | If formulas are displayed, returns TRUE; otherwise FALSE. |
| 9 | If gridlines are displayed, returns TRUE; otherwise FALSE. |
| 10 | If row and column headings are displayed, returns TRUE; otherwise FALSE. |
| 11 | If zeros are displayed, returns TRUE; otherwise FALSE. |
| 12 | Gridline and heading color as a number in the range 1 to 16, corresponding to the colors in the Options Display dialog box; if color is automatic, returns 0. |

Values 13 to 16 for *type_num* return arrays that specify which rows or columns are at the top and left edges of the panes in the window and the widths and heights of those panes.

- The first number in the array corresponds to the first pane, the second number to the second pane, and so on.
- If the edge of the pane occurs at the boundary between rows or columns, the number returned is an integer.
- If the edge of the pane occurs within a row or column, the number returned has a fractional part that represents the fraction of the row or column visible within the pane.
- The numbers can be used as arguments to the SPLIT function to split a window at specific locations.

| *Type_num* | Returns |
|---|---|
| 13 | Leftmost column number of each pane, in a horizontal numeric array. |
| 14 | Top row number of each pane, in a horizontal numeric array. |
| 15 | Number of columns in each pane, in a horizontal numeric array. |
| 16 | Number of rows in each pane, in a horizontal numeric array. |
| 17 | Number indicating the active pane:<br>1 = Upper, left, or upper-left<br>2 = Right or upper-right<br>3 = Lower or lower-left<br>4 = Lower-right |
| 18 | If window has a vertical split, returns TRUE; otherwise FALSE. |
| 19 | If window has a horizontal split, returns TRUE; otherwise FALSE . |
| 20 | If window is maximized, returns TRUE; otherwise FALSE. |
| 21 | Reserved. |
| 22 | If outline symbols are displayed, returns TRUE; otherwise FALSE. |

***Window_text*** is the name of the window that you want information about, as the name appears in the title bar. If *window_text* is omitted, it is assumed to be the active window.

### Examples

If the active window contains the document Macro1, then:

```
GET.WINDOW(1) equals "Macro1"
```

If the title of the active window is Macro1:3, then:

```
GET.WINDOW(2) equals 3
```

In Microsoft Excel for Windows or OS/2, the following macro formula returns the gridline and heading color of REPORT.XLS:

```
GET.WINDOW(12,"REPORT.XLS")
```

In Microsoft Excel for the Macintosh, the following macro formula returns the gridline and heading color of Report Master:

```
GET.WINDOW(12,"Report Master")
```

### Related Functions

| | |
|---|---|
| GET.DOCUMENT | Returns information about a document |
| GET.WORKSPACE | Returns information about the workspace |

*Macro Sheets Only*

## GET.WORKSPACE(type_num)

Returns information about the workspace. Use GET.WORKSPACE in a macro that depends on the status of the workspace, such as the environment, version number, and available memory.

***Type_num*** is a number specifying the type of workspace information you want. The following list shows the *type_num* values and their corresponding results:



| *Type num* | Returns |
|---|---|
| 1 | Name of the environment in which Microsoft Excel is running, as text, followed by the environment's version number. |
| 2 | The version number of Microsoft Excel, as text (for example, "3.0"). |
| 3 | If auto-decimal is set, returns the number of decimals; otherwise, returns 0. |
| 4 | If in R1C1 mode, returns TRUE; if in A1 mode, returns FALSE. |
| 5 | If scroll bars are displayed, returns TRUE; otherwise FALSE. |
| 6 | If the status bar is displayed, returns TRUE; otherwise FALSE. |
| 7 | If the formula bar is displayed, returns TRUE; otherwise FALSE. |
| 8 | If remote DDE requests are enabled, returns TRUE; otherwise FALSE. |
| 9 | Returns the alternate menu key as text; if no alternate menu key is set, returns #N/A. |
| 10 | Number indicating special modes:<br>1 = Data Find<br>2 = Copy<br>3 = Cut<br>If no special mode is set, returns 0. |
| 11 | X position of the Microsoft Excel workspace window, measured in points from the left edge of the screen to the left edge of the window. In Microsoft Excel for the Macintosh, always returns 0. |
| 12 | Y position of the Microsoft Excel workspace window, measured in points from the top edge of the screen to the top edge of the window. In Microsoft Excel for the Macintosh, always returns 0. |
| 13 | Usable workspace width, in points. |
| 14 | Usable workspace height, in points. |

| *Type num* | Returns |
|---|---|
| 15 | Number indicating maximized or minimized status of Microsoft Excel<br>1 = Neither<br>2 = Minimized<br>3 = Maximized<br>Microsoft Excel for the Macintosh always returns 3. |
| 16 | Amount of memory free (in kilobytes). In Microsoft Excel for OS/2, this number does not take into account virtual memory. |
| 17 | Total memory available to Microsoft Excel (in kilobytes). In Microsoft Excel for OS/2, this number does not take into account virtual memory. |
| 18 | If a math coprocessor is present, returns TRUE; otherwise FALSE. |
| 19 | If a mouse is present, returns TRUE; otherwise FALSE. In Microsoft Excel for the Macintosh, always returns TRUE. |
| 20 | If a workgroup is present in the workspace, returns a horizontal array of sheets in the workgroup; otherwise returns #N/A. |
| 21 | If the tool bar is displayed, returns TRUE; otherwise FALSE. |
| 22 | DDE-application-specific error code. |
| 23 | Full path of the default startup directory or folder. |
| 24 | Full path of the alternate startup directory or folder; returns #N/A if no alternate path has been specified. |
| 25 | If Microsoft Excel is set for relative recording, returns TRUE; if set for absolute recording, returns FALSE. |
| 26 | Name of user. |
| 27 | Name of organization. |
| 28 | If Microsoft Excel menus are activated by the alternate menu or help key, returns 1; if Lotus 1-2-3 Help is activated, returns 2. |
| 29 | If alternate navigation keys are enabled, returns TRUE. |

***Related Functions***

| | |
|---|---|
| GET.DOCUMENT | Returns information about a document |
| GET.WINDOW | Returns information about a window |

*Macro Sheets Only*

## GOAL.SEEK(target_cell,target_value, variable_cell)

## GOAL.SEEK?(target_cell,target_value, variable_cell)

Equivalent to the Formula Goal Seek command. Calculates the values necessary to achieve a specific goal. If the goal is an amount returned by a formula, the GOAL.SEEK function calculates values that, when supplied to your formula, cause your formula to return the amount you want. For detailed information on the Formula Goal Seek command, see online Help.

***Target_cell*** corresponds to the Set Cell box in the Formula Goal Seek dialog box and is a reference to the cell containing the formula. If *target_cell* does not contain a formula, Microsoft Excel displays an error message.

***Target_value*** corresponds to the To Value box in the Formula Goal Seek dialog box and is the value you want the formula in *target_cell* to return. This value is called a goal.

***Variable_cell*** corresponds to the By Changing Cell box in the Formula Goal Seek dialog box and is the single cell that you want Microsoft Excel to change so that the formula in *target_cell* returns *target_value*. *Target_cell* must depend on *variable_cell*; if it does not, Microsoft Excel will not be able to find a solution.

*Tip*

You can also use Microsoft Excel Solver to help solve your math equations for optimal values. For more information, see the *Microsoft Excel Solver User's Guide*.

***Example***

Suppose a worksheet has a cell named Binomial containing the formula =(X^3)+(3*X^2)+6 and another cell named X that is empty. The X in Binomial refers to the cell named X. The following macro formulas solve the equation on the active worksheet so that it equals 15:

```
GOAL.SEEK("Binomial",15,"X")
GOAL.SEEK(!Binomial,15,!X)
```

The result is 1.425988576 and is stored in cell X on the active worksheet. The formula in Binomial evaluates to 14.99999734.

*Macro Sheets Only*

## GOTO(reference)

Directs a macro to continue running at the upper-left cell of *reference*. Use GOTO to direct macro execution to another cell or a named range.

***Reference*** is a cell reference or a name that is defined as a reference. *Reference* can be an external reference to another macro sheet. If that macro sheet is not open, GOTO displays a message.

*Tip*

It's often preferable to use IF, ELSE, ELSE.IF, and END.IF instead of GOTO when you want to perform multiple actions based on a condition because the IF method makes your macros more structured.

***Examples***

If A1 contains the #N/A error value, then when the following formula is calculated, the macro branches to C3:

```
IF(ISERROR($A$1),GOTO($C$3),)
```

You can also use macro names with GOTO statements. The following formula branches macro execution to a macro named Compile:

```
GOTO(Compile)
```

Because Compile is a named range, it should not be enclosed in quotation marks.



***Related Function***

| | |
|---|---|
| FORMULA.GOTO | Selects the specified reference or named range |

*Macro Sheets Only*

## GRIDLINES(x_major,x_minor,y_major,y_minor, z_major,z_minor)

## GRIDLINES?(x_major,x_minor,y_major,y_minor, z_major,z_minor)

Equivalent to the Chart Gridlines command. Allows you to turn chart gridlines on and off. For detailed information on the Chart Gridlines command, see online Help.

Arguments are logical values corresponding to the check boxes in the Chart Gridlines dialog box. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box. The default value for all arguments in GRIDLINES is FALSE.

***X_major*** corresponds to the Category (X) Axis: Major Gridlines check box.

***X_minor*** corresponds to the Category (X) Axis: Minor Gridlines check box.

***Y_major*** corresponds to the Value (Y) Axis: Major Gridlines check box. On 3-D charts, *y_major* corresponds to the Series (Y) Axis: Major Gridlines check box.

***Y_minor*** corresponds to the Value (Y) Axis: Minor Gridlines check box. On 3-D charts, *y_minor* corresponds to the Series (Y) Axis: Minor Gridlines check box.

***Z_major*** corresponds to the Value (Z) Axis: Major Gridlines check box (3-D only).

***Z_minor*** corresponds to the Value (Z) Axis: Minor Gridlines check box (3-D only).

*Macro Sheets Only*

## GROUP()

Equivalent to the Format Group command. Creates a single object from several selected objects and returns the object identifier of the group (for example, "Group 5"). Use GROUP to combine a number of objects so that you can move or resize them together. For detailed information on the Format Group command, see online Help.

If no object is selected, only one object is selected, or a group is already selected, GROUP returns the #VALUE! error value and interrupts the macro.

***Related Function***

| | |
|---|---|
| UNGROUP | Separates grouped objects |

## GROWTH(known_y's,known_x's,new_x's,const)

Fits an exponential curve to the data *known_y's* and *known_x's*, and returns the y-values along that curve for the array of *new_x's* values that you specify. For more information on exponential curves, see LOGEST.

***Known_y's*** is the set of y-values you already know in the relationship $y = b*m^x$.

- If the array *known_y's* is in a single column, then each column of *known_x's* is interpreted as a separate variable.
- If the array *known_y's* is in a single row, then each row of *known_x's* is interpreted as a separate variable.
- If any of the numbers in *known_y's* is 0 or negative, GROWTH returns the #NUM! error value.

***Known_x's*** is an optional set of *x*-values that you may already know in the relationship $y = b*m^x$.

- The array *known_x's* can include one or more sets of variables. If only one variable is used, *known_y's* and *known_x's* can be ranges of any shape, as long as they have equal dimensions. If

more than one variable is used, *known_y's* must be a vector (that is, a range with a height of one row or a width of one column).

- If *known_x's* is omitted, it is assumed to be the array {1,2,3,...} that is the same size as *known_y's*.

***New_x's*** are new *x*-values for which you want GROWTH to return corresponding *y*-values.

- *New_x's* must include a column (or row) for each independent variable, just as *known_x's* does. So, if *known_y's* is in a single column, *known_x's* and *new_x's* should have the same number of columns. If *known_y's* is in a single row, *known_x's* and *new_x's* should have the same number of rows.
- If you omit *new_x's*, it is assumed to be the same as *known_x's*.
- If you omit both *known_x's* and *new_x's*, they are assumed to be the array {1,2,3,...} that is the same size as *known_y's*.

***Const*** is a logical value specifying whether to force the constant *b* to equal 1.

- If *const* is TRUE or omitted, *b* is calculated normally.
- If *const* is FALSE, *b* is set equal to 1 and the *m*-values are adjusted so that $y = m^x$.

### Notes

- Formulas that return arrays must be entered as array formulas.
- When entering an array constant for an argument such as *known_x's*, use commas to separate values in the same row and semicolons to separate rows.

### Examples

This example uses the same data as the LOGEST example. The sales for the 11th through the 16th month are 33100, 47300, 69000, 102000, 150000, and 220000 units. Assume that these values are entered into six cells named UnitsSold.

When entered as an array formula, the following formula predicts sales for months 17 and 18 based on sales for the previous six months:

GROWTH(UnitsSold,{11;12;13;14;15;16},{17;18},) equals {320197;468536}

If the exponential trend continues, sales for months 17 and 18 will be 320197 and 468536 units, respectively.

You could use other sequential numbers for the *x*-value arguments, and the predicted sales would be the same. For example, you could use the default value for *known_x's*, {1;2;3;4;5;6}:

GROWTH(UnitsSold,,{7;8},) equals {320197;468536}

### Related Functions

| | |
|---|---|
| LOGEST | Also calculates a regression curve, but returns the parameters of that curve instead of an array of *y*-values along the curve |
| TREND and LINEST | Are similar to GROWTH and LOGEST, but they fit your data to a straight line |

*Macro Sheets Only*

## HALT(*cancel_close*)

Stops all macros from running. Use HALT instead of RETURN to prevent a macro from returning to the macro that called it.

***Cancel_close*** is a logical value that specifies whether a macro sheet, when encountering the HALT function in an Auto_Close macro, is closed.

- If *cancel_close* is TRUE, Microsoft Excel halts the macro and prevents the document from being closed.
- If *cancel_close* is FALSE or omitted, Microsoft Excel halts the macro and allows the document to be closed.
- If *cancel_close* is specified in a macro that is not an Auto_Close macro, it is ignored and the HALT function simply stops the current macro.

### Note

You can prevent an Auto_Close or Auto_Open macro from running by holding down the SHIFT key while opening or closing the document.



### Examples

If A1 contains the #N/A error value, then when the following formula is calculated, the macro halts:

IF(ISERROR(A1),HALT(),GOTO(D4))

The following formula at the end of an Auto_Close macro ends the macro and prevents the document from being closed:

HALT(TRUE)

### Related Functions

| | |
|---|---|
| BREAK | Interrupts a FOR–NEXT, WHILE–NEXT, or FOR.CELL–NEXT loop |
| RETURN | Stops the currently running macro or returns values to a custom function |

*Macro Sheets Only*

## HELP(*file_text!topic_num*)

Starts or activates Help and displays the custom Help topic specified by *file_text!topic_num*. Use HELP with custom Help text files to create your own Help system, which can be used just like the built-in Microsoft Excel Help.

### Notes

- Microsoft Excel version 3.0 for Windows or OS/2 no longer supports the use of Help files in the text file format for custom Help. If you need to convert your existing Help files to the Windows or OS/2 Help file format, you can obtain a conversion kit by contacting your software vendor or Microsoft Corporation.
- If you want to create custom Help with improved functionality, such as glossary terms and jump topics, in Microsoft Excel version 3.0 for Windows or OS/2, you can purchase the Windows or OS/2 Software Development Kit from your software vendor or Microsoft Corporation.
- In Microsoft Excel for the Macintosh, custom Help files are plain text files or text files with line breaks.

***File_text!topic_num*** is a reference to a topic in a text file, in the form "filename!topic_number".

- *File_text!topic_num* must be given as quoted text.
- If *file_text!topic_num* is omitted, HELP displays the Microsoft Excel Help index.
- The *topic_num* in the Help file must be preceded by an asterisk.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula displays the Help topic numbered 101 in the file HELP.TXT:

HELP("HELP.TXT!101")

If the custom Help file is not in the current directory, specify the full pathname along with the name of the file. For example:

HELP("C:\EXCEL\HELP.TXT!101")

In Microsoft Excel for the Macintosh, the following macro formula displays the Help topic numbered 101 in the file Custom Help:

HELP("Custom Help!101")

If the custom Help file is not in the current folder, specify the full pathname along with the name of the file. For example:

HELP("Hard Disk:Excel:Help:Custom Help!101")

In a Microsoft Excel file, Help text looks like this:

| | A | B |
|---|---|---|
| 1 | *35 Using the new menus | |
| 2 | This worksheet automatically replaces | |
| 3 | Microsoft Excel's menus with menus designed to | |
| 4 | run our company's reporting system. You can | |
| 5 | get help on any of the menu commands by | |
| 6 | pressing Shift+F1 and selecting the command. | |
| 7 | | |
| 8 | *36 Retrieve Report command | |
| 9 | This command allows you to open one of the | |
| 10 | preformatted reports from the Reports | |
| 11 | directory. This command will only open one | |
| 12 | file in that directory. To open more than one | |
| 13 | file, use the Open Any command. | |

For information on custom Help topics, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

Macro Sheets Only

## HIDE()

Equivalent to the Window Hide command (full menus). Hides the active window. For detailed information on the Window Hide command, see online Help.

### Tip

Hiding windows can speed up your macros. Hidden windows can be activated with the ACTIVATE function. You can continue to use functions that refer to specific sheets, such as FORMULA and the GET functions, even when those sheets are hidden.

### Related Function

| | |
|---|---|
| UNHIDE | Displays a hidden document |

Macro Sheets Only

## HIDE.OBJECT(object_id_text,hide)

Hides or displays the specified object.

***Object_id_text*** is the name and number, or number alone, of the object, as text, as it appears in the reference area when the object is selected. The name of the object is also the text returned by the CREATE.OBJECT function, so *object_id_text* can be a reference to a cell containing CREATE.OBJECT. To give the name of more than one object, use the following format for *object_id_text*:

"oval 3,text 2,arc 5"

If *object_id_text* is omitted, the function operates on all selected objects. If no object is selected or if the object specified by *object_id_text* does not exist, HIDE.OBJECT returns the #VALUE! error value.

***Hide*** is a logical value that specifies whether to hide or display the specified object. If *hide* is TRUE or omitted, Microsoft Excel hides the object; if FALSE, Microsoft Excel displays the object.

### Note

Objects are not automatically selected after they are unhidden.

### Examples

The following macro formula hides the selected object:

HIDE.OBJECT(,TRUE)

The following macro formula displays the object named Oval 3:

HIDE.OBJECT("oval 3",FALSE)

The following macro formula displays the three specified objects:

HIDE.OBJECT("oval 3,text 2,arc 5",FALSE)

### Related Functions

| | |
|---|---|
| CREATE.OBJECT | Creates the specified object |
| DISPLAY | Controls how an object is displayed |

Macro Sheets Only

## HLINE(num_columns)

Scrolls the active window by a specific number of columns. Returns the #VALUE! error value if the active document is a chart.

***Num_columns*** is the number of columns in the active worksheet or macro sheet you want to scroll horizontally.

- If *num_columns* is positive, HLINE scrolls to the right.
- If *num_columns* is negative, HLINE scrolls to the left.

### Related Functions

| | |
|---|---|
| HPAGE | Horizontally scrolls the active window one full window at a time |
| HSCROLL | Horizontally scrolls a document by percentage or by column number |
| VLINE | Vertically scrolls the active window by rows |



## HLOOKUP(lookup_value,table_array, row_index_num)

Searches the top row of an array for a particular value, and returns the value in the indicated cell. Use HLOOKUP when your comparison values are located in a row across the top of a table of data and you want to look down a specified number of rows. Use VLOOKUP when your comparison values are located in a column to the left or right of the data you want to find.

***Lookup_value*** is the value to be found in the first row of the table. *Lookup_value* can be a value, a reference, or a quoted text string.

***Table_array*** is a table of information from which data is looked up. Use a reference to a range or a range name, such as Database.

- The values in the first row of *table_array* can be text, numbers, or logical values.
- The values in the first row of *table_array* must be placed in ascending order: ...–2, –1, 0, 1, 2,... , A–Z, FALSE, TRUE; otherwise HLOOKUP may not give the correct value.
- Uppercase and lowercase text are equivalent.
- You can put values in ascending order by selecting the values, choosing the Data Sort command, and selecting Ascending.

***Row_index_num*** is the row number in *table_array* from which the matching value should be returned. A *row_index_num* of 1 returns the first row value in *table_array*, a *row_index_num* of 2 returns the second row value in *table_array*, and so on. If *row_index_num* is less than 1, HLOOKUP returns the #VALUE! error value; if *row_index_num* is greater than the number of rows on *table_array*, HLOOKUP returns the #REF! error value.

### Notes

- If HLOOKUP can't find *lookup_value*, it uses the largest value that is less than or equal to *lookup_value*.
- If *lookup_value* is smaller than the smallest value in the first row of *table_array*, HLOOKUP returns the #N/A error value.

### Examples

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | | Axles | Bearings | Bolts | Gears | Tires | Wheels |
| 2 | | 4 | 3 | 6 | 2 | 5 | 1 |
| 3 | | 5 | 4 | 7 | 3 | 6 | 2 |
| 4 | | 6 | 5 | 8 | 4 | 7 | 3 |
| 5 | | 7 | 6 | 9 | 5 | 8 | 4 |
| 6 | | 8 | 7 | 10 | 6 | 9 | 5 |
| 7 | | 9 | 8 | 11 | 7 | 10 | 6 |
| 8 | | 10 | 9 | 12 | 8 | 11 | 7 |
| 9 | | 11 | 10 | 13 | 9 | 12 | 8 |
| 10 | | 12 | 11 | 14 | 10 | 13 | 9 |

In the worksheet above:

HLOOKUP("Tires",B1:G10,3) equals 6

HLOOKUP("Bolts",B1:G10,5) equals 9

HLOOKUP("Bearings",B1:G10,7) equals 8

*Table_array* can also be an array constant:

HLOOKUP(3,{1,2,3;"a","b","c";"d","e","f"},2) equals "c"

### Related Functions

| | |
|---|---|
| INDEX | Uses an index to choose a value from a reference or array |
| LOOKUP and MATCH | Look up values in a reference or array |
| VLOOKUP | Looks in the first column of an array and moves across the row to return the value of a cell |

## HOUR(serial_number)

Returns the hour corresponding to *serial_number*. The hour is given as an integer, ranging from 0 (12:00 A.M.) to 23 (11:00 P.M.).

***Serial_number*** is the date-time code used by Microsoft Excel for date and time calculations. You can give *serial_number* as text, such as "16:48:00" or "4:48:00 PM", instead of as a number. The text is automatically converted to a serial number. For more information about *serial_number*, see NOW.

### Note

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

### Examples

HOUR(0.7) equals 16

HOUR(29747.7) equals 16

HOUR("3:30:30 PM") equals 15

### Related Functions

| | |
|---|---|
| NOW | Returns the serial number of the current date and time |
| YEAR, MONTH, DAY, WEEKDAY, MINUTE, and SECOND | Convert serial numbers into years, months, days, days of the week, minutes, or seconds |

Macro Sheets Only

## HPAGE(num_windows)

Horizontally scrolls the active window one window at a time. Use HPAGE to change the displayed area of a worksheet or macro sheet.

***Num_windows*** specifies the number of windows to scroll the active window horizontally. A window is defined as the number of visible columns. If three columns are visible in the window, HPAGE scrolls in increments of three columns.

- If *num_windows* is positive, HPAGE scrolls to the right.
- If *num_windows* is negative, HPAGE scrolls to the left.

### Example

The following macro formula scrolls the active window 14 windows to the right:

```
HPAGE(14)
```

### Related Functions

| | |
|---|---|
| HLINE | Horizontally scrolls the active window by columns |
| HSCROLL | Horizontally scrolls a document by percentage or by column number |
| VPAGE | Vertically scrolls the active window one full window at a time |

Macro Sheets Only

## HSCROLL(position,col_logical)

Horizontally scrolls the active document by percentage or by column number.

***Position*** specifies the column you want to scroll to. *Position* can be an integer representing the column number or a fraction or percentage representing the horizontal position of the column in the document. If *position* is 0, HSCROLL scrolls your document to its leftmost edge. If *position* is 1, HSCROLL scrolls your document to its rightmost edge.

***Col_logical*** is a logical value specifying how the function scrolls.

- If *col_logical* is TRUE, HSCROLL scrolls the document to column *position*.
- If *col_logical* is FALSE or omitted, then HSCROLL scrolls the document to the horizontal position represented by the fraction *position*.

### Notes

- To scroll to a specific column *n*, either use HSCROLL(*n*,TRUE) or use HSCROLL(*n*/256). To scroll to column 38, for example, use HSCROLL(38,TRUE) or HSCROLL(38/256).
- If you are recording a macro and move the scroll box several times in a row, the recorder only records the final location of the scroll box, omitting any intermediate steps. Remember that scrolling does not change the active cell or the selection.



### Examples

The following macro formulas all scroll the document to column 128 (column DX in A1 format), 50% of the way across the document:

```
HSCROLL(128,TRUE)
HSCROLL(50%)
HSCROLL(0.5,FALSE)
HSCROLL(128/256)
```

Also see the last example for VSCROLL.

### Related Functions

| | |
|---|---|
| HLINE | Horizontally scrolls the active window by columns |
| HPAGE | Horizontally scrolls the active window one full window at a time |
| VSCROLL | Vertically scrolls a document by percentage or by row number |

## IF(logical_test,value_if_true,value_if_false)

Form 1: Worksheets and macro sheets

Returns one value if *logical_test* evaluates to TRUE, and another value if it evaluates to FALSE.

There are two forms of the IF function. This form can be used on worksheets and macro sheets. The second form of the function can only be used on macro sheets in conjunction with the ELSE, ELSE.IF, and END.IF functions.

Use IF to conduct conditional tests on values and formulas and to branch based on the result of that test. The outcome of the test determines the value returned by the IF function.

***Logical_test*** is any value or expression that can be evaluated to TRUE or FALSE.

***Value_if_true*** is the value that is returned if *logical_test* is TRUE. If *logical_test* is TRUE and *value_if_true* is omitted, TRUE is returned.

***Value_if_false*** is the value that is returned if *logical_test* is FALSE. If *logical_test* is FALSE and *value_if_false* is omitted, FALSE is returned.

### Notes

- Up to seven IF functions can be nested as *value_if_true* and *value_if_false* arguments to construct more elaborate tests. See the last example below.
- If you are using IF in a macro, *value_if_true* and *value_if_false* can also be GOTO functions, other macros, or action-taking functions.

  For example, the following formula is allowed in a macro:

  ```
  IF(Number>10,GOTO(Large),GOTO(Small))
  ```

  In the above example, if Number is greater than 10, then *logical_test* is TRUE, the *value_if_true* statement is evaluated, and the macro function GOTO(Large) is run. If Number is less than or equal to 10, then *logical_test* is FALSE, *value_if_false* is evaluated, and the macro function GOTO(Small) is run.
- When the *value_if_true* and *value_if_false* arguments are evaluated, IF returns the value returned by those statements. In the example above, if the number is not greater than 10, TRUE is still returned if the second GOTO statement is successful.
- If any of the arguments to IF are arrays, every element of the array is evaluated when the IF statement is carried out. If some of the *value_if_true* and *value_if_false* arguments are action-taking functions, all of the actions are taken. For example, the following macro formula runs both ALERT functions:

  ```
  IF({TRUE,FALSE},ALERT("One",2),ALERT("Two",2))
  ```

### Examples

In the following example, if the value referred to by the name File is equal to Chart, *logical_test* is TRUE and the macro function NEW(2) is carried out, otherwise, *logical_test* is FALSE and NEW(1) is carried out.

```
IF(File="Chart",NEW(2),NEW(1))
```

The worksheet below shows actual and predicted expense for January through April.

| | A | B | C |
|---|---|---|---|
| 1 | | Actual Expenses | Predicted Expenses |
| 2 | January | 1500 | 900 |
| 3 | February | 500 | 900 |
| 4 | March | 500 | 925 |
| 5 | April | 600 | 925 |

You could write a macro to check whether you are over budget for a particular month, generating text for a message with the following formulas:

IF(B2>C2,"Over Budget","OK") equals "Over Budget"

IF(B3>C3,"Over Budget","OK") equals "OK"

Suppose you want to assign letter grades to numbers referenced by the name Average:

| If Average is | Then return |
|---|---|
| Greater than 89 | A |
| From 80 to 89 | B |
| From 70 to 79 | C |
| From 60 to 69 | D |
| Less than 60 | F |

You could use the following nested IF function:

IF(Average>89,"A",IF(Average>79,"B",IF(Average>69,"C",IF(Average>59,"D","F"))))

In the example above, the second IF statement is also the *value_if_false* argument to the first IF statement. Similarly, the third IF statement is the *value_if_false* argument to the second IF statement. For example, if the first *logical_test* (Average>89) is TRUE, "A" is returned. If the first *logical_test* is FALSE, the second IF statement is evaluated, and so on.

### Related Functions

| | |
|---|---|
| AND | Returns TRUE if all its arguments are TRUE |
| FALSE | Returns the logical value FALSE |
| NOT | Reverses the logic of its argument |
| OR | Returns TRUE if any argument is TRUE |
| TRUE | Returns the logical value TRUE |

*Macro Sheets Only*

## IF(*logical_test*)

**Form 2: Macro sheets only**

Used with ELSE, ELSE.IF, and END.IF to control which formulas in a macro are executed. There are two forms of the IF function. This form can be used only on macro sheets; use it when you want your macro to branch to a particular set of functions based on the outcome of a logical test.

***Logical_test*** is a logical value that IF uses to determine which functions to carry out next—that is, where to branch.

- If *logical_test* is TRUE, Microsoft Excel carries out the functions between the IF function and the next ELSE, ELSE.IF, or END.IF function. Instructions between ELSE.IF or ELSE and END.IF are not carried out.
- If *logical_test* is FALSE, Microsoft Excel immediately branches to the next ELSE.IF, ELSE, or END.IF function.

### Note

Form 1 of the IF function can also be used on macro sheets.

### Tips

- Use IF with ELSE, ELSE.IF, and END.IF when you want to perform multiple actions based on a condition. This method is preferable to using GOTO because it makes your macros more structured.
- If your macro ends with an error at a cell containing this form of the IF function, make sure there is a corresponding END.IF function.

### Example

The following macro asks for the age of someone applying for an insurance policy. Three things can happen: If the person is under 18, the insurance agent must attach a rider for children to the policy. If the person is between 18 and 34 (younger than 35), the macro calls a subroutine macro to display a new menu. If the person is over 34, the insurance agent must use another form. This macro assumes that the subroutines AttachChildRider and InsuranceMenu are defined on other parts of the macro sheet.



| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | ApplyForInsurancePolicy | |
| 4 | Age | =INPUT("Enter applicant's age",1) | |
| 5 | | =IF(Age<18) | |
| 6 | ChildRider | =ALERT("This policy is for adults. Do you wa | |
| 7 | | =IF(ChildRider) | |
| 8 | | =AttachChildRider(Age) | Runs subroutine |
| 9 | | =END.IF() | |
| 10 | | =ELSE.IF(Age<35) | |
| 11 | | =InsuranceMenu(Age) | Runs subroutine |
| 12 | | =ELSE() | |
| 13 | | =ALERT("Use application form 2",2) | |
| 14 | | =END.IF() | |
| 15 | | =RETURN() | |

The complete formula in cell B6 is:

```
=ALERT("This policy is for adults. Do you want to attach a rider for children?",1)
```

### Tip

You can indent formulas in a macro. To indent a formula, type as many spaces as you want between the equal sign and the first letter of the formula. The following macro shows how the preceding macro looks when you indent formulas in it.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | ApplyForInsurancePolicy | |
| 4 | Age | =INPUT("Enter applicant's age",1) | |
| 5 | | =IF(Age<18) | |
| 6 | ChildRider | = ALERT("This policy is for adults. Do you | |
| 7 | | = IF(ChildRider) | |
| 8 | | = AttachChildRider(Age) | Runs subroutine |
| 9 | | = END.IF() | |
| 10 | | =ELSE.IF(Age<35) | |
| 11 | | = InsuranceMenu(Age) | Runs subroutine |
| 12 | | = ELSE() | |
| 13 | | = ALERT("Use application form 2",2) | |
| 14 | | =END.IF() | |
| 15 | | =RETURN() | |

For more information on indenting formulas in macros, see Chapter 19, "Designing and Writing a Command Macro," in *the Microsoft Excel User's Guide*.

For more examples of IF, see ARGUMENT, FILES, UNHIDE, and WHILE.

### Related Functions

| | |
|---|---|
| CHOOSE | Chooses a value from a list of values or runs a macro from a list of macros |
| ELSE | Specifies what action to take if the preceding IF function returns FALSE |
| ELSE.IF | Specifies a logical test to perform if the preceding IF or ELSE.IF function returns FALSE |
| END.IF | Ends a block of functions associated with the preceding IF or ELSE.IF function |

## INDEX(*reference,row_num,column_num,area_num*)

**Form 1: Reference form**

Returns the reference of the cell at the intersection of a particular row and column. If the reference is made up of multiple selections, you can pick the selection to look in.

The INDEX function has two forms: reference and array. The reference form always returns a reference; the array form always returns a value or an array of values.

***Reference*** is a reference to one or more cell ranges.

- If you are entering a multiple selection for *reference*, enclose *reference* in parentheses. For an example of using INDEX with a multiple selection, see the fifth example below.
- If each area in *reference* contains only one row or column, the *row_num* or *column_num* argument, respectively, is optional. For example, for a single row *reference*, use INDEX(*reference*,,*column_num*).

***Row_num*** is the number of the row in *reference* from which to return a reference.

***Column_num*** is the number of the column in *reference* from which to return a reference.

***Area_num*** selects a range in *reference* from which to return the intersection of *row_num* and *column_num*. The first area selected or entered is numbered 1, the second is 2, and so on. If *area_num* is omitted, INDEX uses area 1.

For example, if *reference* describes the cells (A1:B4,D1:E4,G1:H4), then *area_num* 1 is the range A1:B4, *area_num* 2 is the range D1:E4, and *area_num* 3 is the range G1:H4.

After *reference* and *area_num* have selected a particular range, *row_num* and *column_num* select a particular cell: *row_num* 1 is the first row in the range, *column_num* 1 is the first column, and so on. The reference returned by INDEX is the intersection of *row_num* and *column_num*.

If you set *row_num* or *column_num* to 0, INDEX returns the reference for the entire column or row, respectively.

### Notes

- *Row_num*, *column_num*, and *area_num* must point to a cell within *reference*; otherwise, INDEX returns the #REF! error value.
- The result of the INDEX function is a reference and is interpreted as such by other formulas. Depending on the formula, the return value of INDEX may be used as a reference or as a value. For example, the formula CELL("width",INDEX(A1:B2,1,2)) is equivalent to CELL("width",B1). The CELL function uses the return value of INDEX as a cell reference. On the other hand, a formula such as 2*INDEX(A1:B2,1,2) translates the return value of INDEX into the number in cell B1.

### Examples

In the worksheet below, the range A2:C6 is named Fruit, the range A8:C11 is named Nuts, and the range A1:C11 is named Stock.

| | A | B | C |
|---|---|---|---|
| 1 | | Price | Count (lbs.) |
| 2 | Apples | $0.69 | 40 |
| 3 | Bananas | $0.34 | 38 |
| 4 | Lemons | $0.55 | 15 |
| 5 | Oranges | $0.25 | 25 |
| 6 | Pears | $0.59 | 40 |
| 7 | | | |
| 8 | Almonds | $2.80 | 10 |
| 9 | Cashews | $3.55 | 16 |
| 10 | Peanuts | $1.25 | 20 |
| 11 | Walnuts | $1.75 | 12 |

INDEX(Fruit,2,3) equals the reference C3, containing 38

INDEX(Nuts,1,1) equals the reference A8, containing "Almonds"

INDEX(Fruit,2,4) equals #REF!, because the *column_num* argument (4) is out of range

INDEX(Stock,1,2) equals the reference B2, containing "Price"

INDEX((A1:C6,A8:C11),2,2,2) equals the reference B9, containing $3.55

SUM(INDEX(Stock,0,3,1)) equals SUM(C1:C11) equals 216

SUM(B2:INDEX(Fruit,5,2)) equals SUM(B2:B6) equals 2.42

### Related Functions

| | |
|---|---|
| CHOOSE | Chooses a value from a list of values or runs a macro from a list of macros |
| HLOOKUP | Looks in the first row of an array and moves down the column to return the value of a cell |
| LOOKUP and MATCH | Look up values in a reference or array |
| VLOOKUP | Looks in the first column of an array and moves across the row to return the value of a cell |

## INDEX(*array,row_num,column_num*)

**Form 2: Array form**

Returns the value of an element in an array, selected by the row and column number indexes.

The INDEX function has two forms: reference and array. The reference form always returns a reference; the array form always returns a value or array of values. Use the array form when the first argument to INDEX is an array constant.

***Array*** is a range of cells entered as an array.

***Row_num*** selects the row in *array* from which to return a value. If *row_num* is omitted, *column_num* is required.



***Column_num*** selects the column in *array* from which to return a value. If *column_num* is omitted, *row_num* is required.

- If both the *row_num* and *column_num* arguments are used, INDEX returns the value in the cell at the intersection of *row_num* and *column_num*.
- If *array* contains only one row or column, the corresponding *row_num* or *column_num* argument is optional.
- If *array* has more than one row and more than one column, and only *row_num* or *column_num* is used, INDEX returns an array of the entire row or column in *array*.
- If you set *row_num* or *column_num* to 0, INDEX returns the array of values for the entire column or row, respectively. To use values returned as an array, enter the INDEX function as an array in a horizontal array of cells. Arrays are entered by pressing CTRL+SHIFT+ENTER (in Microsoft Excel for Windows or OS/2) or COMMAND+ENTER (in Microsoft Excel for the Macintosh).

### Note

*Row_num* and *column_num* must point to a cell within *array*; otherwise, INDEX returns the #REF! error value.

### Examples

INDEX({1,2;3,4},2,2) equals 4

If entered as an array formula, then:

INDEX({1,2;3,4},0,2) equals {2;4}

### Related Functions

| | |
|---|---|
| CHOOSE | Chooses a value from a list of values or runs a macro from a list of macros |
| HLOOKUP | Looks in the first row of an array and moves down the column to return the value of a cell |
| LOOKUP and MATCH | Look up values in a reference or array |
| VLOOKUP | Looks in the first column of an array and moves across the row to return the value of a cell |

## INDIRECT(*ref_text,a1*)

Returns the reference specified by *ref_text*. References are immediately evaluated to display their contents. Use INDIRECT to get the value stored in a cell indicated by a reference in another cell.

***Ref_text*** is a reference to a cell that contains an A1 reference, an R1C1 reference, or a name defined as a reference. If *ref_text* is not a valid cell reference, INDIRECT returns the #REF! error value.

***A1*** is a logical value that specifies what type of reference is contained in the cell *ref_text*.

- If *a1* is TRUE or omitted, *ref_text* is interpreted as an A1-style reference.
- If *a1* is FALSE, *ref_text* is interpreted as an R1C1-style reference.

### Examples

If cell A1 contains the text "B2", and cell B2 contains the value 1.333, then:

INDIRECT($A$1) equals 1.333

If the workspace is set to display R1C1-style references, cell R1C1 contains R2C2, and cell R2C2 contains the value 1.333, then:

INT(INDIRECT(R1C1,FALSE)) equals 1

If B3 contains the text "George", and a cell defined as George contains the value 10, then:

INDIRECT($B$3) equals 10.

### Related Functions

| | |
|---|---|
| OFFSET | Returns a reference offset from a given reference |
| TEXTREF | Converts text to a reference |

## INFO(*type_num*)

Returns information about the current operating environment.

***Type_num*** is text specifying what type of information you want returned.

| *Type_num* | Returns |
|---|---|
| "directory" | Pathname of the current directory or folder. |
| "memavail" | Amount of memory available, in bytes. In Microsoft Excel for OS/2, this number does not take into account virtual memory. |
| "numfile" | Number of active worksheets. |
| "osversion" | Current operating system version, as text. |
| "recalc" | Current recalculation mode; returns "Automatic" or "Manual". |
| "release" | Version of Microsoft Excel, as text. |
| "system" | Name of the operating environment:<br>Macintosh = "mac"<br>OS/2 = "pcos2"<br>Windows = "pcdos" |
| "totmem" | Total memory available, including memory already in use, in bytes. In Microsoft Excel for OS/2, this number does not take into account virtual memory. |
| "memused" | Amount of memory being used for data. |

### Examples

The following macro formula returns 2 if two worksheets are currently open:

```
INFO("numfile")
```

The following macro formula returns "pcos2" if the current operating environment is OS/2:

```
INFO("system")
```

### Related Functions

| | |
|---|---|
| DIRECTORY | Returns the current directory or allows you to specify a new directory |
| GET.DOCUMENT | Returns information about a document |
| GET.WORKSPACE | Returns information about the workspace |

*Macro Sheets Only*

## INITIATE(*app_text,topic_text*)

Opens a DDE channel to an application, and returns the number of the open channel. Once you have opened a channel to another application with INITIATE, you can use EXECUTE and SEND.KEYS to control the other application from a Microsoft Excel macro. (SEND.KEYS is available only with Microsoft Excel for Windows or OS/2.) If INITIATE is successful, it returns the number of the open channel. All the subsequent DDE macro functions use this number to specify the channel. If INITIATE is unsuccessful, it returns #N/A.

### Important

Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

***App_text*** is the DDE name of the application with which you want to begin a DDE session, enclosed in quotation marks. The form of *app_text* depends on the application you are accessing. The DDE name of Microsoft Excel, for example, is "Excel".

***Topic_text*** describes something, such as a document or a record in a database, in the application that you are accessing; the form of *topic_text* depends on the application you are accessing. Microsoft Excel accepts the names of the current documents as *topic_text*, as well as the name "System".

For more information on DDE, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*.

### Notes

- You can specify an instance of an application by appending the application's task ID number to the *app_text* argument. If you start an application by using the EXEC function, EXEC returns the task ID number for that instance of the application.
- If more than one instance of an application is running and you do not specify which instance you would like to open a channel to, INITIATE displays a dialog box from which you can choose the instance you want. You can prevent this dialog box from appearing by disabling or redirecting errors with the ERROR function.



### Examples

The following macro formula opens a channel to the document named MEMO in the application named WORD:

```
INITIATE("WORD","MEMO")
```

In Microsoft Excel for Windows or OS/2, the following macro starts Word for Windows, opens a document, executes some formatting commands, and terminates the link between the two documents:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | FromWord | |
| 4 | | =EXEC("C:\SALES.DOC",3) | Starts Word, loads file |
| 5 | Channel | =INITIATE("WINWORD","C:\SALES.DOC") | Initiates channel to Word |
| 6 | | =EXECUTE(Channel,"[DOWN 3]") | Selects 3 lines down |
| 7 | | =EXECUTE(Channel,"[Copy]") | Copies the selection |
| 8 | | =APP.ACTIVATE(,FALSE) | Activates Microsoft Excel |
| 9 | | =SELECT("R2C3") | Selects C2 |
| 10 | | =PASTE() | Pastes text |
| 11 | | =TERMINATE(Channel) | Terminates Word channel |
| 12 | | =RETURN() | |

In Microsoft Excel for the Macintosh, the following macro starts Word, executes some commands, and terminates the link between the two documents:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | FromWord | |
| 4 | | =EXEC("HardDisk:Microsoft Word") | Starts Word |
| 5 | Channel | =INITIATE("Microsoft Word","Untitled") | Initiates channel to Word |
| 6 | | =EXECUTE(Channel,"V") | Pastes Clipboard |
| 7 | | =EXECUTE(Channel,"P~") | Sends the print command |
| 8 | | =APP.ACTIVATE(,FALSE) | Activates Microsoft Excel |
| 9 | | =CREATE.OBJECT(6,"R9C2",,,"R15C4",,) | Creates a text box |
| 10 | | =PASTE() | Pastes text into box |
| 11 | | =TERMINATE(Channel) | Terminates Word channel |
| 12 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| POKE | Sends data to another application |
| REQUEST | Returns data from another application |
| TERMINATE | Closes a channel to another application |

*Macro Sheets Only*

## INPUT(*message_text,type_num,title_text,default,x_pos,y_pos*)

Displays a dialog box for user input. Returns the information entered in the dialog box. Use INPUT to display a simple dialog box for the user to enter information to be used in a macro.

The dialog box has an OK and a Cancel button. If you choose OK, INPUT returns the default value specified or the value typed in the edit box. If you choose Cancel, INPUT returns FALSE.

***Message_text*** is the text to be displayed in the dialog box. *Message_text* must be enclosed in quotation marks.

***Type_num*** is a number specifying the type of data to be entered:

| *Type_num* | Data type |
|---|---|
| 0 | Formula |
| 1 | Number |
| 2 | Text |
| 4 | Logical |
| 8 | Reference |
| 16 | Error |
| 64 | Array |

You can also use the sum of the allowable data types for *type_num*. For example, for an input box that can accept text or numbers, set *type_num* equal to 3, the sum of 1 and 2, the number and text type specifiers. If *type_num* is omitted, it is assumed to be 2.

- If *type_num* is 0, INPUT returns the formula in the form of text, for example, "=2*PI()/360".
- To enter a formula, include an equal sign at the beginning of the formula; otherwise the formula is returned as quoted text.
- If the formula contains references, they are returned as R1C1-style references, for example, "=RC[-1]*(1+R1C1)".

- If *type_num* is 8, INPUT returns an absolute reference to the specified cells.
- If you enter a single-cell reference in the edit box, the value in that cell is returned by the INPUT function.
- If the information entered in the edit box is not of the correct data type, Microsoft Excel attempts to convert it to the specified type. If it can't, it displays an error message. For information on data types, see "Converting Data Types," earlier in this book.

***Title_text*** is text specifying a title to be displayed in the title bar of the dialog box. If *title_text* is omitted, it is assumed to be "Input".

***Default*** specifies a value to be shown in the edit box when the dialog box is initially displayed. If *default* is omitted, the edit box is empty.

***X pos, y pos*** specify the horizontal and vertical position, in points, of the dialog box. A point is 1/72nd of an inch. If either or both arguments are omitted, the dialog box is centered in that direction.

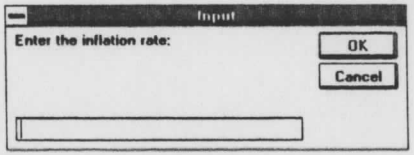
### Note

Relative references entered in the INPUT dialog box are relative to the active cell at the time the INPUT function is calculated. If you are using the reference entered into the dialog box in a cell other than the active cell, it may not refer to the cells you intend it to. For example, if the active cell is A3 and you enter the formula "=A1+A2" in an INPUT dialog box, intending to add the values in those cells, and then use the FORMULA function to enter the formula in cell B3, the formula in cell B3 will read "=B1+B2" because you gave a relative reference.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula displays the dialog box shown below:

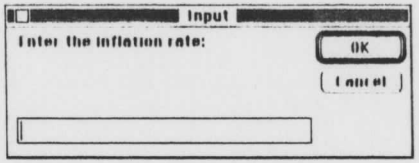
```
INPUT("Enter the inflation rate:",1)
```




If you then enter 12%, INPUT returns the value 0.12.

In Microsoft Excel for the Macintosh, the following macro formula displays the dialog box shown below:

```
INPUT("Enter the inflation rate:",1)
```




If you then enter 12%, INPUT returns the value 0.12.

If the active cell is C2 and you enter the formula =B2*(1+$A$1) in response to the following macro formula:

```
INPUT("Enter your monthly increase formula:",0)
```

INPUT returns "=RC[-1]*(1+R1C1)"

If you select the range $A$2:$A$8 in the INPUT dialog box:

```
REFTEXT(INPUT("Please make your selection.",8))
```

returns R2C1:R8C1

### Related Functions

| | |
|---|---|
| ALERT | Displays a dialog box with a message |
| DIALOG.BOX | Displays a custom dialog box |



*Macro Sheets Only*

## INSERT(*shift_num*)
## INSERT?(*shift_num*)

Equivalent to the Edit Insert command. Inserts a blank cell or range of cells or pastes cells from the Clipboard into a worksheet. Shifts the selected cells to accommodate the new ones. The size and shape of the inserted range are the same as those of the current selection. For detailed information on the Edit Insert command, see online Help.

***Shift_num*** is a number from 1 to 4 specifying which way to shift the cells. If an entire row or column is selected, *shift_num* is optional (and is ignored if specified). If *shift_num* is omitted, Microsoft Excel shifts cells in the logical direction based on the selection.

| *Shift_num* | Direction |
|---|---|
| 1 | Shift cells right |
| 2 | Shift cells down |
| 3 | Shift entire row |
| 4 | Shift entire column |

### Note

If you have just cut or copied information to the Clipboard, INSERT performs both an insert and a paste operation. First, Microsoft Excel inserts new blank cells into the worksheet; then, Microsoft Excel pastes information from the Clipboard into the newly inserted cells. If you have used the INSERT function in macros written for Microsoft Excel version 2.2 or earlier, make sure you consider this feature when you use your old macros with Microsoft Excel version 3.0.

### Related Functions

| | |
|---|---|
| COPY | Copies information to the Clipboard |
| CUT | Defines the selection that will be moved when you use PASTE |
| EDIT.DELETE | Removes cells from a worksheet |
| PASTE | Pastes copied or cut data |

## INT(*number*)

Rounds a number down to the nearest integer.

***Number*** is the real number you want to round down to an integer.

### Examples

INT(8.9) equals 8

INT(-8.9) equals -9

The following formula returns the decimal part of a positive real number in cell A1:

```
A1-INT(A1)
```

### Related Functions

| | |
|---|---|
| MOD | Gives the remainder from division |
| ROUND | Rounds a number to a specified number of digits |
| TRUNC | Truncates a number to an integer |

## IPMT(*rate,per,nper,pv,fv,type*)

Returns the interest payment for a given period for an investment based on periodic, constant payments and a constant interest rate.

For a more complete description of the arguments in IPMT and for more information on annuity functions, see PV.

***Rate*** is the interest rate per period.

***Per*** is the period for which you want to find the interest, and must be in the range 1 to *nper*.

***Nper*** is the total number of payment periods in an annuity.

***Pv*** is the present value, or the lump-sum amount that a series of future payments is worth right now.

***Fv*** is the future value, or a cash balance you want to attain after the last payment is made. If *fv* is omitted, it is assumed to be 0 (the future value of a loan, for example, is 0).

***Type*** is the number 0 or 1 and indicates when payments are due. If *type* is omitted, it is assumed to be 0.

| Set *type* equal to | If payments are due |
|---|---|
| 0 | At the end of the period |
| 1 | At the beginning of the period |

### Notes

- Make sure that you are consistent about the units you use for specifying *rate* and *nper*. If you make monthly payments on a four-year loan at 12% annual interest, use 12%/12 for *rate* and 4*12 for *nper*. If you make annual payments on the same loan, use 12% for *rate* and 4 for *nper*.
- For all the arguments, cash you pay out, such as deposits to savings, is represented by negative numbers; cash you receive, such as dividend checks, is represented by positive numbers.

### Examples

The following formula calculates the interest due in the first month of a three-year $8,000 loan at 10% annual interest:

IPMT(0.1/12,1,36,8000) equals –$66.67

The following formula calculates the interest due in the last year of a three-year $8,000 loan at 10% annual interest, where payments are made yearly:

IPMT(0.1,3,3,8000) equals –$292.45

### Related Functions

| | |
|---|---|
| FV | Returns the future value of an investment |
| NPER | Returns the number of payments for an investment |
| PMT | Returns the periodic total payment for an investment |
| PPMT | Returns the payment on the principal for an investment for a given period |
| PV | Returns the present value of an investment |
| RATE | Returns the interest rate per period of an investment |

## IRR(values,guess)

Returns the internal rate of return for a series of periodic cash flows represented by the numbers in *values*. These cash flows do not have to be even, as they would be for an annuity. The internal rate of return is the interest rate received for an investment consisting of payments (negative values) and income (positive values) that occur at regular periods.

***Values*** is an array or a reference to cells that contain numbers for which you want to calculate the internal rate of return.

- *Values* must contain at least one positive value and one negative value to calculate the internal rate of return.
- IRR uses the order of *values* to interpret the order of cash flows. Be sure to enter your payment and income values in the sequence you want.
- *Values* containing text, logical, or empty cells, are ignored.

***Guess*** is a number that you guess is close to the result of IRR.

- Microsoft Excel uses an iterative technique for calculating IRR. Starting with *guess*, IRR cycles through the calculation until the result is accurate within 0.00001%. If IRR can't find a result that works after 20 tries, the #NUM! error value is returned.
- In most cases you do not need to provide *guess* for the IRR calculation. If omitted, *guess* is assumed to be 0.1 (10%).
- If IRR gives the #NUM! error value, or if the result is not close to what you expected, try again with a different value for *guess*.

### Examples

Suppose you want to start a restaurant business. You estimate it will cost $70,000 to start the business and expect to net the following income in the first five years: $12,000, $15,000, $18,000, $21,000, and $26,000.



| | A | B | C |
|---|---|---|---|
| 1 | Initial Investment | ($70,000) | |
| 2 | Year 1 Income | $12,000 | |
| 3 | Year 2 Income | $15,000 | |
| 4 | Year 3 Income | $18,000 | |
| 5 | Year 4 Income | $21,000 | |
| 6 | Year 5 Income | $26,000 | |

To calculate the investment's internal rate of return after four years:

IRR(B1:B5) equals –2.12%

To calculate the internal rate of return after five years:

IRR(B1:B6) equals 8.66%

To calculate the internal rate of return after two years, you need to include a *guess*:

IRR(B1:B3,–10%) equals –44.35%

### Note

IRR is closely related to NPV, the net present value function. The rate of return calculated by IRR is the interest rate corresponding to a zero net present value. The following formula demonstrates how NPV and IRR are related:

NPV(IRR(B1:B6),B1:B6) equals 3.60E–08

(Within the accuracy of the IRR calculation, the value 3.60E–08 is effectively 0.)

### Related Functions

| | |
|---|---|
| MIRR | Returns the internal rate of return where positive and negative cash flows are financed at different rates |
| NPV | Returns the net present value of an investment based on cash flows that do not have to be constant |
| RATE | Returns the interest rate per period of an investment |

## IS Functions

## ISBLANK(value)
## ISERR(value)
## ISERROR(value)
## ISLOGICAL(value)
## ISNA(value)
## ISNONTEXT(value)
## ISNUMBER(value)
## ISREF(value)
## ISTEXT(value)

This section describes the nine worksheet functions used for testing the type of a value or reference.

Each of these functions, referred to collectively as the IS functions, checks the type of *value* and returns TRUE or FALSE depending on the outcome. For example, the ISBLANK function returns the logical value TRUE if *value* is a reference to an empty cell; otherwise it returns FALSE.

***Value*** is the blank (empty cell), error, logical, text, number, or reference value you want to test.

| Function | Returns TRUE if |
|---|---|
| ISBLANK | *Value* refers to an empty cell. |
| ISERR | *Value* refers to any error value except #N/A. |
| ISERROR | *Value* refers to any error value (#N/A, #VALUE!, #REF!, #DIV/0!, #NUM!, #NAME?, or #NULL!). |
| ISLOGICAL | *Value* refers to a logical value. |
| ISNA | *Value* refers to the #N/A (value not available) error value. |
| ISNONTEXT | *Value* refers to any item that is not text. (Note that this function returns TRUE if *value* refers to a blank cell.) |
| ISNUMBER | *Value* refers to a number. |
| ISREF | *Value* refers to a reference. |
| ISTEXT | *Value* refers to text. |

### Notes

- The *value* arguments to the IS functions are not converted. For example, in most other functions where a number is required, the text value "19" is converted to the number 19. However, in the formula ISNUMBER("19"), "19" is not converted from a text value and the ISNUMBER function returns FALSE.
- If *value* is an array, the IS functions test only the first element in the array. This is true even if the formulas containing the IS functions are array formulas.
- The IS functions are useful in formulas and macros for testing the outcome of a calculation. When combined with the IF function, they provide a method for locating errors in formulas (see the following examples).

### Examples

ISLOGICAL(TRUE) equals TRUE

ISLOGICAL("TRUE") equals FALSE

ISNUMBER(4) equals TRUE

| | A | B | C | D |
|---|---|---|---|---|
| 1 | | Gold | Silver | Oil |
| 2 | Region1 | $350.89 | $5.99 | $20.22 |
| 3 | Region2 | $330.92 | #REF! | $21.00 |
| 4 | Region3 | #N/A | $6.71 | $20.50 |

In the preceding worksheet:

ISBLANK(C1) equals FALSE

ISERROR(C3) equals TRUE

ISNA(C3) equals FALSE

ISNA(B4) equals TRUE

ISERR(B4) equals FALSE

ISNUMBER(B3) equals TRUE (if the $330.92 was entered as a number and not as text)

ISREF(Region1) equals TRUE (if Region1 is defined as a range name)

ISTEXT(A2) equals TRUE

On another worksheet, suppose you want to calculate the average of the range A1:A4, but you can't be sure that the cells contain numbers. The formula AVERAGE(A1:A4) returns the error value #DIV/0! if A1:A4 does not contain any numbers. To allow for this case, you could use the following formula to locate potential errors:

IF(ISERROR(AVERAGE(A1:A4)),"No Numbers",AVERAGE(A1:A4))

The following worksheet balances a checkbook. Each cell in column G adds deposits to or subtracts withdrawals from the cell above it (the previous balance). For example, cell G151 equals G150+E151–F151. To display an empty cell instead of a zero if you haven't made a deposit or withdrawal, use the following formula:

IF(AND(ISBLANK(E151),ISBLANK(F151)),"",G150+E151–F151)

| | C | D | E | F | G |
|---|---|---|---|---|---|
| 1 | Date | Check No. | Deposits | Withdrawals | Balance |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| 147 | 20-Jan | 2035 | | $50.45 | $1,045.00 |
| 148 | 21-Jan | | $125.33 | | $1,170.33 |
| 149 | 21-Jan | 2036 | | $46.77 | $1,123.56 |
| 150 | 22-Jan | 2037 | | $88.92 | $1,034.64 |
| 151 | | | | | |

### Related Functions

GET.CELL — Returns information about the formatting, location, or contents of a cell

TYPE — Returns a number indicating the data type of a value

Macro Sheets Only

## JUSTIFY()

Equivalent to the Format Justify command (full menus). Rearranges the text in a range so that it fills the range evenly. For detailed information on the Format Justify command, see online Help.

### Related Function

ALIGNMENT — Aligns or wraps text in cells



Macro Sheets Only

## LAST.ERROR()

Returns the reference of the cell where the last macro sheet error occurred. If no error has occurred, LAST.ERROR returns the #N/A error value. Use LAST.ERROR in conjunction with the ERROR function to quickly locate errors.

### Related Function

ERROR — Specifies an action to take if an error occurs while a macro is running

## LEFT(text,num_chars)

Returns the first (or leftmost) character or characters in a text string.

***Text*** is the text string containing the characters you want to extract.

***Num_chars*** specifies how many characters you want LEFT to return.

- *Num_chars* must be greater than zero.
- If *num_chars* is greater than the length of *text*, LEFT returns all of *text*.
- If *num_chars* is omitted, it is assumed to be 1.

### Examples

LEFT("Sale Price",4) equals "Sale"

If A1 contains "Sweden", then:

LEFT(A1) equals "S"

Suppose a hardware store assigns stock numbers to its products. The first three characters of the stock number identify the product vendor. On a worksheet, assume that Stock refers to a cell containing a stock number and that VendorTable is a sorted two-column table containing all the vendor numbers in the first column and the corresponding vendor names in the second.

You can use the LEFT function to obtain the first three characters of the stock number (a vendor number) and the VLOOKUP function to return the vendor's name as shown in the following formula:

VLOOKUP(LEFT(Stock,3),VendorTable,2)

### Related Functions

MID — Extracts characters based on a starting position and number of characters

RIGHT — Extracts the rightmost characters from a text value

Macro Sheets Only

## LEGEND(logical)

Equivalent to the Chart Add Legend and Chart Delete Legend commands. Adds a legend to or removes a legend from a chart. For detailed information on the Chart Add Legend and Chart Delete Legend commands, see online Help.

***Logical*** is a logical value specifying which command LEGEND is equivalent to.

- If *logical* is TRUE or omitted, LEGEND is equivalent to the Chart Add Legend command.
- If *logical* is FALSE, LEGEND is equivalent to the Chart Delete Legend command.
- If *logical* is FALSE and the active chart has no legend, LEGEND takes no action.

### Related Function

FORMAT.LEGEND — Determines the position and orientation of the legend on a chart

## LEN(text)

Returns the number of characters in a text string.

***Text*** is the text whose length you want to find. Spaces count as characters.

### Examples

LEN("Phoenix, AZ") equals 11

LEN("") equals 0

The following macro formula starts a FOR–NEXT loop on a macro sheet. The number of iterations is equal to the length of a string named CheckText:

FOR("Counter",1,LEN(CheckText))

Also see the third example in CODE.

### Related Functions

| | |
|---|---|
| EXACT | Checks to see if two text values are identical |
| SEARCH | Finds one text value within another (not case sensitive) |

## LINEST(known_y's,known_x's,const,stats)

Uses the "least squares" method to calculate a straight line that best fits your data and returns an array that describes the line. The equation for the line is:

$y = m_1x_1 + m_2x_2 + \cdots + b$ or $y = mx + b$

where the dependent *y*-value is a function of the independent *x*-values. The *m*-values are coefficients corresponding to each *x*-value, and *b* is a constant value. Note that *y*, *x*, and *m* can be vectors. The array that LINEST returns is $\{m_n,m_{n-1},\ldots,m_1,b\}$. LINEST can also return additional regression statistics.

***Known_y's*** is the set of *y*-values you already know in the relationship $y = mx + b$.

- If the array *known_y's* is in a single column, then each column of *known_x's* is interpreted as a separate variable.
- If the array *known_y's* is in a single row, then each row of *known_x's* is interpreted as a separate variable.

***Known_x's*** is an optional set of *x*-values that you already know in the relationship $y = mx + b$.

- The array *known_x's* can include one or more sets of variables. If only one variable is used, *known_y's* and *known_x's* can be ranges of any shape, as long as they have equal dimensions. If more than one variable is used, *known_y's* must be a vector (that is, a range with a height of one row or a width of one column).
- If *known_x's* is omitted, it is assumed to be the array {1,2,3,...} that is the same size as *known_y's*.

***Const*** is a logical value specifying whether to force the constant *b* to equal 0.

- If *const* is TRUE or omitted, *b* is calculated normally.
- If *const* is FALSE, *b* is set equal to 0 and the *m*-values are adjusted to fit $y = mx$.

***Stats*** is a logical value specifying whether to return additional regression statistics.

- If *stats* is TRUE, LINEST returns the additional regression statistics, so the returned array is $\{m_n,m_{n-1},\ldots,m_1,b;se_n,se_{n-1},\ldots,se_1,se_b;r^2,se_y;F,df;ss_{reg},ss_{resid}\}$.
- If *stats* is FALSE or omitted, LINEST returns only the *m*-coefficients and the constant *b*.

The additional regression statistics are:

| Statistic | Description |
|---|---|
| $se_1,se_2,\ldots,se_n$ | The standard error values for the coefficients $m_1,m_2,\ldots,m_n$. |
| $se_b$ | The standard error value for the constant *b* ($se_b$ = #NA when *const* is FALSE). |
| $r^2$ | The coefficient of determination. Compares estimated and actual *y*-values, and ranges in value from 0 to 1. If it is 1, there is a perfect correlation in the sample—there is no difference between the estimated *y*-value and the actual *y*-value. At the other extreme, if the coefficient of determination is 0, the regression equation is not helpful in predicting a *y*-value. For information on how $r^2$ is calculated, see the notes below. |



| Statistic | Description |
|---|---|
| $se_y$ | The standard error for the *y* estimate. |
| *F* | The *F* statistic, or the *F*-observed value. Use the *F* statistic to determine whether the observed relationship between the dependent and independent variables occurs by chance. |
| df | The degrees of freedom. Use the degrees of freedom to help you find *F*-critical values in a statistical table. Compare the values you find in the table to the *F* statistic returned by LINEST to determine a confidence level for the model. |
| $ss_{reg}$ | The regression sum of squares. |
| $ss_{resid}$ | The residual sum of squares. |

The following illustration shows the order in which the additional regression statistics are returned:

| | | | | | |
|---|---|---|---|---|---|
| $m_n$ | $m_{n-1}$ | . . . | $m_2$ | $m_1$ | $b$ |
| $se_n$ | $se_{n-1}$ | . . . | $se_2$ | $se_1$ | $se_b$ |
| $r^2$ | $se_y$ | | | | |
| $F$ | df | | | | |
| $ss_{reg}$ | $ss_{resid}$ | | | | |

### Notes

- You can describe any straight line with the slope and the *y*-intercept:

  Slope (*m*):
  To find the slope of a line, often written as *m*, take two points on the line, $(x_1,y_1)$ and $(x_2,y_2)$; the slope is equal to $(y_2 - y_1)/(x_2 - x_1)$.

  *Y*-intercept (*b*):
  The *y*-intercept of a line, often written as *b*, is the value of *y* at the point where the line crosses the y-axis.

  The equation of a straight line is $y = mx + b$. Once you know the values of *m* and *b*, you can calculate any point on the line by plugging the *y*- or *x*-value into that equation. You can also use the TREND function. For more information, see TREND later in this book.

- When you have only one independent *x*-variable, you can obtain the slope and *y*-intercept values directly by using the following formulas:

  Slope:
  INDEX(LINEST(*known_y's*,*known_x's*),1)

  *Y*-intercept:
  INDEX(LINEST(*known_y's*,*known_x's*),2)

- The accuracy of the line calculated by LINEST depends on the degree of scatter in your data. The more linear the data, the more accurate the LINEST model. LINEST uses the method of least squares for determining the best fit for the data. When you have only one independent *x*-variable, the calculations for *m* and *b* are based on the following formulas:

$$m = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum(x^2)) - (\sum x)^2}$$

$$b = \frac{(\sum y)(\sum(x^2)) - (\sum x)(\sum xy)}{n(\sum(x^2)) - (\sum x)^2}$$

- The line- and curve-fitting functions LINEST and LOGEST can calculate the best straight line or exponential curve that fits your data. However, you have to decide which of the two results best fits your data. To see if the calculated line or curve fits your data:

  1. Calculate TREND(*known_y's*,*known_x's*)—for a straight line—and GROWTH(*known_y's*, *known_x's*)—for an exponential curve. These functions, without the *new_x's* argument, return an array of *y*-values predicted along that line or curve at your actual data points.
  2. Compare the predicted values with the actual values. You may want to chart them both for a visual comparison.

- In regression analysis, Microsoft Excel calculates for each point the squared difference between the *y*-value estimated for that point and its actual *y*-value. The sum of these squared differences is called the residual sum of squares. Microsoft Excel then calculates the sum of the squared differences between the actual *y*-values

and the average of the *y*-values, which is called the total sum of squares (regression sum of squares + residual sum of squares). The smaller the residual sum of squares is compared with the total sum of squares, the larger the value of the coefficient of determination, $r^2$, which is an indicator of how well the equation resulting from the regression analysis explains the relationship among the variables.

- Formulas that return arrays must be entered as array formulas.
- When entering an array constant such as *known_x's* as an argument, use commas to separate values in the same row and semicolons to separate rows. Separator characters may be different depending on your country settings.
- You should note that the *y*-values predicted by the regression equation may not be valid if they are outside the range of the *y*-values you used to determine the equation.

### Examples

#### Example 1: Slope and Y-intercept

LINEST({1,9,5,7},{0,4,2,3}) equals {2,1}, the slope = 2 and *y*-intercept = 1

#### Example 2: Simple Linear Regression

Suppose a small business has sales of $3,100, $4,500, $4,400, $5,400, $7,500, and $8,100 during the first six months of the fiscal year. Assuming that the values are entered in the range B2:B7, you can use the following simple linear regression model to estimate sales for the ninth month.

SUM(LINEST(B2:B7)*{9,1}) equals SUM({1000,2000}*{9,1}) equals $11,000.

In general, SUM({*m*,*b*}*{*x*,1}) equals $mx + b$, the estimated *y*-value for a given *x*-value. You can also use the TREND function.

#### Example 3: Multiple Linear Regression

Suppose a commercial developer is considering purchasing a group of small office buildings in an established business district.

The developer can use multiple linear regression analysis to estimate the value of an office building in a given area based on the following variables:

| Variable | Refers to the |
|---|---|
| $y$ | Assessed value of the office building |
| $x_1$ | Floor space in square feet |
| $x_2$ | Number of offices |
| $x_3$ | Number of entrances |
| $x_4$ | Age of the office building in years |

This example assumes that a straight-line relationship exists between each of the independent variables ($x_1$, $x_2$, $x_3$, and $x_4$) and the dependent variable ($y$), the value of office buildings in the area.

The developer randomly chooses a sample of 11 office buildings from a possible 1,500 office buildings, and obtains the following data:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | x1<br>Floor Space | x2<br>Offices | x3<br>Entrances | x4<br>Age | y<br>Value |
| 2 | 2,310 | 2 | 2 | 20 | $142,000 |
| 3 | 2,333 | 2 | 2 | 12 | $144,000 |
| 4 | 2,356 | 3 | 1.5 | 33 | $151,000 |
| 5 | 2,379 | 3 | 2 | 43 | $150,000 |
| 6 | 2,402 | 2 | 3 | 53 | $139,000 |
| 7 | 2,425 | 4 | 2 | 23 | $169,000 |
| 8 | 2,448 | 2 | 1.5 | 99 | $126,000 |
| 9 | 2,471 | 2 | 2 | 34 | $142,900 |
| 10 | 2,494 | 3 | 3 | 23 | $163,000 |
| 11 | 2,517 | 4 | 4 | 55 | $169,000 |
| 12 | 2,540 | 2 | 3 | 22 | $149,000 |

"Half an entrance" means an entrance for deliveries only. When entered as an array, the following formula:

LINEST(E2:E12,A2:D12,TRUE,TRUE)

returns the following output:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 14 | -234.23716 | 2553.21066 | 12529.7682 | 27.6413874 | 52317.8305 |
| 15 | 13.2680115 | 530.669152 | 400.066838 | 5.42937404 | 12237.3616 |
| 16 | 0.99674799 | 970.578463 | #N/A | #N/A | #N/A |
| 17 | 459.753674 | 6 | #N/A | #N/A | #N/A |
| 18 | 1732393319 | 5652135.32 | #N/A | #N/A | #N/A |



The multiple regression equation, $y = m_1*x_1 + m_2*x_2 + m_3*x_3 - m_4*x_4 + b$, can now be obtained using the values from row 14:

$y = 27.64*x_1 + 12,530*x_2 + 2,553*x_3 - 234.24*x_4 + 52,318$

The developer can now estimate the assessed value of an office building in the same area which has 2,500 square feet, three offices, and two entrances, and which is 25 years old, by using the following equation:

$y = 27.64*2500 + 12,530*3 + 2,553*2 - 234.24*25 + 52318 = \$158,261$

You can also use the TREND function to calculate this value. For more information, see TREND.

#### Example 4: Using the F and R-squared Statistics

In the previous example, the coefficient of determination, or $r^2$, is 0.99675 (see cell A16 in the output for LINEST), which would indicate a strong relationship between the dependent variables and the sale price. You can use the *F* statistic to determine whether these results, with such a high $r^2$ value, occurred by chance.

Assume for the moment that in fact there is no relationship among the variables, but that you have drawn a rare sample of 11 office buildings which causes the statistical analysis to demonstrate a strong relationship. The probability of erroneously concluding that there is a relationship is called Alpha.

There is a relationship among the variables if the *F*-observed statistic is greater than the *F*-critical value. The *F*-critical value can be obtained by referring to a table of *F*-critical values in many statistics textbooks. To read the table, assume a single-tailed test, use an Alpha value of 0.05, and for the degrees of freedom (abbreviated in most tables as *v*1 and *v*2) use $v1 = k = 4$ and $v2 = n - (k + 1) = 11 - (4 + 1) = 6$, where $k$ is the number of variables in the regression analysis and $n$ is the number of data points. The *F*-critical value is 4.53.

The *F*-observed value is 459.753674 (cell A17), which is substantially greater than the *F*-critical value of 4.53. Therefore, the regression equation is useful in predicting the assessed value of office buildings in this neighborhood.

#### Example 5: Calculating the t-Statistics

Another hypothesis test will determine whether each slope coefficient is useful in estimating the assessed value of an office building in example 4. For example, to test the age coefficient for statistical significance, divide -234.24 (age slope coefficient) by 13.268 (the estimated standard error of age coefficients in cell A15). The following is the *t*-observed value:

$t = m_4 + se_4 = -234.24 + 13.268 = -17.7$

If you consult a table in a statistics manual, you will find that *t*-critical, single tail, with 6 degrees of freedom and Alpha = 0.05 is 1.94 . Since the absolute value of *t*, 17.7, is greater than 1.94, age is an important variable when estimating the assessed value of an office building. Each of the other dependent variables can be tested for statistical significance in a similar manner. The following are the *t*-observed values for each of the dependent variables:

| | |
|---|---|
| Floor space | 5.1 |
| Number of offices | 31.3 |
| Number of entrances | 4.8 |
| Age | -17.7 |

These values all have an absolute value greater than 1.94; therefore, all the variables used in the regression equation are useful in predicting the assessed value of office buildings in this area.

#### Related Functions

| | |
|---|---|
| GROWTH and LOGEST | Are similar to TREND and LINEST, but fit an exponential curve to your data instead of a straight line |
| TREND | Calculates a straight line, but returns an array of predicted *y*-values instead of the parameters of the line |

Macro Sheets Only

## LINKS(*document_text,type_num*)

Returns, as a horizontal array of text values, the names of all worksheets referred to by external references in the document specified. Use LINKS with OPEN.LINKS to open supporting documents.

***Document_text*** is the name of a document, including its path. If *document_text* is omitted, LINKS operates on the active document. If the document specified by *document_text* is not open, LINKS returns the #N/A error value.

***Type_num*** is a number from 1 to 6 specifying the type of linked documents to return.

| *Type_num* | Returns |
|---|---|
| 1 or omitted | Microsoft Excel link |
| 2 | DDE link |
| 3 | Reserved |
| 4 | Outgoing NewWave link |
| 5 | Publisher |
| 6 | Subscriber |

### Notes

- If the active document contains no external references, LINKS returns the #N/A error value.
- With the INDEX function, you can select individual worksheet names from the array for use in other functions that take document names as arguments.
- The names of the documents are always returned in alphabetical order. If supporting sheets are open, LINKS returns the names of the documents; if supporting sheets are closed, LINKS includes the full pathname of each document.
- If *type_num* is 5 or 6, LINKS returns a two-row array in which the first row contains the edition name and the second row contains the reference.

### Examples

If a chart named Chart1 is open and contains links to worksheets Data1 and Data2, and the LINKS function shown below is entered as an array into a two-cell horizontal range:

LINKS("Chart1") equals "Data1" in the first cell of the range and "Data2" in the second cell.

In Microsoft Excel for Windows or OS/2, if the chart named VARIANCE.XLC is open and contains data series that refer to worksheets named BUDGET.XLS and ACTUAL.XLS, then:

OPEN.LINKS(LINKS("VARIANCE.XLS")) opens BUDGET.XLS and ACTUAL.XLS.

In Microsoft Excel for the Macintosh, if the worksheet named Sales 1991 is open and contains references to the worksheets West Sales, South Sales, and East Sales, then;

OPEN.LINKS(LINKS("Sales 1991"))

opens West Sales, South Sales, and East Sales.

### Related Functions

| | |
|---|---|
| CHANGE.LINK | Changes supporting worksheet links |
| GET.LINK.INFO | Returns information about a link |
| OPEN.LINKS | Opens specified supporting documents |
| UPDATE.LINK | Updates a link |

Macro Sheets Only

## LIST.NAMES( )

Equivalent to choosing the Formula Paste Name command and selecting the Paste List option. Lists all names (except hidden names) defined on your worksheet. LIST.NAMES also lists the cells to which the names refer, whether a macro corresponding to a particular name is a command macro or a custom function, and the shortcut key for each command macro. For detailed information on the Formula Paste Name command, see online Help.

### Notes

- If the current selection is a single cell, or four or more columns wide, LIST.NAMES pastes all four types of information about worksheet names. The first column contains cell names. The second column contains the corresponding cell references. The third column contains the number 1 if the name refers to a custom function, the number 2 if it refers to a command macro, or 0 if it refers to anything else. The fourth column lists the shortcut keys for command macros.
- If the selection includes fewer than four columns, LIST.NAMES omits the information that would have been pasted into the other columns.
- When you use LIST.NAMES, Microsoft Excel completely replaces the contents of the cells it pastes into.

### Example

The following macro scrolls to an unused section of the active worksheet, pastes a list of names and information about the names defined on that worksheet, and prints the list.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **PrintNames** | |
| 4 | | =SELECT.SPECIAL(11) | Selects last cell with data |
| 5 | | =SELECT(OFFSET(ACTIVE.CELL(),1,1)) | Selects cell below and right |
| 6 | | =LIST.NAMES() | Lists name information |
| 7 | | =SET.PRINT.AREA() | Sets the print area |
| 8 | | =PRINT(1,,,1,FALSE,FALSE,1,FALSE,1) | |
| 9 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| GET.DEF | Returns a name matching a definition |
| GET.NAME | Returns the definition of a name |
| NAMES | Returns a horizontal text array of all the names that are defined on a document |

## LN(*number*)

Returns the natural logarithm of a number. Natural logarithms are based on the constant e (2.718281828459O4).



***Number*** is the positive real number for which you want the natural logarithm.

LN is the inverse of the EXP function.

### Examples

LN(86) equals 4.454347

LN(2.7182818) equals 1

LN(EXP(3)) equals 3

EXP(LN(4)) equals 4

### Related Functions

| | |
|---|---|
| EXP | Returns e raised to a given power |
| LOG | Returns the logarithm of a number to a specified base |
| LOG10 | Returns the base-10 logarithm of a number |

## LOG(*number,base*)

Returns the logarithm of a number to the base you specify.

***Number*** is the positive real number for which you want the logarithm.

***Base*** is the base of the logarithm. If *base* is omitted, it is assumed to be 10.

### Examples

LOG(10) equals 1

LOG(8,2) equals 3

LOG(86,2.7182818) equals 4.454347

### Related Functions

| | |
|---|---|
| EXP | Returns e raised to a given power |
| LN | Returns the natural logarithm of a number |
| LOG10 | Returns the base-10 logarithm of a number |

## LOG10(number)

Returns the base-10 logarithm of a number.

**Number** is the positive real number for which you want the base-10 logarithm.

### Examples

LOG10(86) equals 1.934498451

LOG10(10) equals 1

LOG10(1E5) equals 5.00E+00

LOG10(10^5) equals 5

### Related Functions

| | |
|---|---|
| EXP | Returns e raised to a given power |
| LN | Returns the natural logarithm of a number |
| LOG | Returns the logarithm of a number to a specified base |

## LOGEST(known_y's,known_x's,const,stats)

Calculates an exponential curve that fits your data, and returns an array that describes the curve. The equation for the curve is

$y = (b*(m_1\text{^}x_1)*(m_2\text{^}x_2)*\cdots)$ or $y = b*m\text{^}x$

where the dependent *y*-value is a function of the independent *x*-values. The *m*-values are bases corresponding to each exponent *x*-value, and *b* is a constant value. Note that *y*, *x*, and *m* can be vectors. The array that LOGEST returns is $\{m_n,m_{n-1},\ldots,m_1,b\}$.

**Known_y's** is the set of *y*-values you already know in the relationship $y = b*m\text{^}x$.

- If the array *known_y's* is in a single column, then each column of *known_x's* is interpreted as a separate variable.
- If the array *known_y's* is in a single row, then each row of *known_x's* is interpreted as a separate variable.

**Known_x's** is an optional set of *x*-values that you already know in the relationship $y = b*m\text{^}x$.

- The array *known_x's* can include one or more sets of variables. If only one variable is used, *known_y's* and *known_x's* can be ranges of any shape, as long as they have equal dimensions. If more than one variable is used, *known_y's* must be a vector (that is, a range with a height of one row or width of one column).
- If *known_x's* is omitted, it is assumed to be the array {1,2,3,...} that is the same size as *known_y's*.

**Const** is a logical value specifying whether to force the constant *b* to equal 1.

- If *const* is TRUE or omitted, *b* is calculated normally.
- If *const* is FALSE, *b* is set equal to 1, and the *m*-values are fit to $y = m\text{^}x$.

**Stats** is a logical value specifying whether to return additional regression statistics.

- If *stats* is TRUE, LOGEST returns the additional regression statistics, so the returned array is $\{m_n,m_{n-1},\ldots,m_1,b;se_n,se_{n-1},\ldots,se_1,se_b;r^2,se_y;F,df;ss_{reg},ss_{resid}\}$.
- If *stats* is FALSE or omitted, LOGEST returns only the *m*-coefficients and the constant *b*.

For more information on the additional regression statistics, see LINEST.

### Notes

- The more a plot of your data resembles an exponential curve, the better the calculated line will fit your data. Like LINEST, LOGEST returns an array of values that describes a relationship among the values, but LINEST fits a straight line to your data; LOGEST fits an exponential curve. For more information, see LINEST earlier in this book.
- When you have only one independent *x*-variable, you can obtain the *m* and *b* values directly by using the following formulas:

  *m*:
  INDEX(LOGEST(*known_y's*,*known_x's*),1)

  *b*:
  INDEX(LOGEST(*known_y's*,*known_x's*),2)



You can use the $y = b*m\text{^}x$ equation to predict future values of *y*, but Microsoft Excel provides the GROWTH function to do this for you. For more information, see GROWTH later in this book.

- Formulas that return arrays must be entered as array formulas.
- When entering an array constant such as *known_x's* as an argument, use commas to separate values in the same row and semicolons to separate rows. Separator characters may be different depending on your country setting.
- You should note that the *y*-values predicted by the regression equation may not be valid if they are outside the range of *y*-values you used to determine the equation.

### Example

After 10 months of sluggish sales, a company experiences an exponential growth in sales after putting a new product on the market. In the subsequent six months, sales increased to 33,100, 47,300, 69,000, 102,000, 150,000, and 220,000 units per month. Assume that these values are entered into six cells named UnitsSold. When entered as an array formula:

LOGEST(UnitsSold,{11;12;13;14;15;16},TRUE,TRUE)

generates the following output:

| | D | E | F |
|---|---|---|---|
| 1 | 1.46327563 | 495.30477 | |
| 2 | 0.0026334 | 0.03583428 | |
| 3 | 0.99980862 | 0.01101631 | |
| 4 | 20896.8011 | 4 | |
| 5 | 2.53601883 | 0.00048544 | |

$y = b*m_1\text{^}x_1$ or using the values from cells D1:E1:
$y = 495.3 * 1.4633^x$

You can estimate sales for future months by substituting the month number for *x* in this equation, or you can use the GROWTH function. For more information, see GROWTH.

You can use the additional regression statistics (cells D2:E5) to determine how useful the equation is for predicting future values.

### Important

The methods you use to test an equation using LOGEST are similar to the methods for LINEST. However, the additional statistics LOGEST returns are based on the following linear model:

$\ln y = x_1 \ln m_1 + \cdots + x_n \ln m_n + \ln b$

You should keep this in mind when you evaluate the additional statistics, especially the $se_i$ and $se_b$ values, which should be compared to $\ln m_i$ and $\ln b$, not to $m_i$ and *b*. For more information, consult an advanced statistics manual.

### Related Functions

| | |
|---|---|
| GROWTH | Also calculates an exponential curve, but returns an array of predicted *y*-values instead of the parameters of the curve |
| TREND and LINEST | Are similar to GROWTH and LOGEST, but fit a straight line instead of an exponential curve to your data |

## LOOKUP(lookup_value,lookup_vector,result_vector)

**Form 1: Vector form**

The LOOKUP function has two forms, vector and array. This section discusses the vector form; the next section discusses the array form.

A vector is an array that contains only one row or one column. The vector form of LOOKUP looks in a vector for a value, moves to the corresponding position in a second vector, and returns this value. Use this form of the LOOKUP function when you want to be able to specify the range that contains the values you want to match. The other form of LOOKUP automatically looks in the first column or row.

***Lookup_value*** is a value that LOOKUP searches for in the first vector. *Lookup_value* can be a number, text, a logical value, or a name or reference that refers to a value.

***Lookup_vector*** is a range that contains only one row or one column. The values in *lookup_vector* can be text, numbers, or logical values.

#### *Important*

The values must be placed in ascending order: . . . ,–2, –1, 0, 1, 2, . . . , A–Z, FALSE, TRUE; otherwise, LOOKUP may not give the correct value. Uppercase and lowercase text are equivalent.

***Result_vector*** is a range that contains only one row or column. It should be the same size as *lookup_vector*.

- If LOOKUP can't find the *lookup_value*, it matches the largest value in *lookup_vector* that is less than or equal to *lookup_value*.
- If *lookup_value* is smaller than the smallest value in *lookup_vector*, LOOKUP gives the #N/A error value.

#### Examples

| | A | B | C |
|---|---|---|---|
| 1 | Frequency | Color | |
| 2 | 4.14234 | red | |
| 3 | 4.19342 | orange | |
| 4 | 5.17234 | yellow | |
| 5 | 5.77343 | green | |
| 6 | 6.38987 | blue | |
| 7 | 7.31342 | violet | |

In the worksheet above:

LOOKUP(4.91,A2:A7,B2:B7) equals "orange"

LOOKUP(5.00,A2:A7,B2:B7) equals "orange"

LOOKUP(7.66,A2:A7,B2:B7) equals "violet"

LOOKUP(7.66E–14,A2:A7,B2:B7) equals #N/A, because 7.66E–14 is less than the smallest value in the *lookup_vector* A2:A7

#### Related Functions

| | |
|---|---|
| HLOOKUP | Looks in the first row of an array and moves down the column to return the value of a cell |
| INDEX | Uses an index to choose a value from a reference or array |
| VLOOKUP | Looks in the first column of an array and moves across the row to return the value of a cell |

## LOOKUP(lookup_value,array)

**Form 2: Array form**

The LOOKUP function has two forms, vector and array. This section discusses the array form; the previous section discusses the vector form.

The array form of LOOKUP looks in the first row or column of an array for the specified value, moves down or across to the last cell, and returns the value of the cell. Use this form of LOOKUP when the values you want to match are in the first row or column of the array. Use the other form of LOOKUP when you want to be able to specify the location of the column or row.

#### *Tip*

In general, it's best to use the HLOOKUP or VLOOKUP function instead of the array form of LOOKUP. This form of LOOKUP is provided for compatibility with other spreadsheet programs.

***Lookup_value*** is a value that LOOKUP searches for in an array. *Lookup_value* can be a number, text, a logical value, or a name or reference that refers to a value.

- If LOOKUP can't find the *lookup_value*, it uses the largest value in the array that is less than or equal to *lookup_value*.
- If *lookup_value* is smaller than the smallest value in the first row or column (depending on the array dimensions), LOOKUP returns the #N/A error value.

***Array*** is a range of cells that contains text, numbers, or logical values that you want to compare with *lookup_value*.



The array form of LOOKUP is very similar to the HLOOKUP and VLOOKUP functions. The difference is that HLOOKUP searches for *lookup_value* in the first row, VLOOKUP searches in the first column, and LOOKUP searches according to the dimensions of *array*.

- If *array* is square, or covers an area that is wider than it is tall (more columns than rows), LOOKUP searches for *lookup_value* in the first row.
- If *array* is taller than it is wide (more rows than columns), LOOKUP searches in the first column.
- HLOOKUP and VLOOKUP allow you to index down or across, but LOOKUP always selects the last value in the row or column.

#### *Important*

The values must be placed in ascending order: …,–2, –1, 0, 1, 2, …, A-Z, FALSE, TRUE; otherwise, LOOKUP may not give the correct value. Uppercase and lowercase are equivalent.

#### Examples

LOOKUP("C",{"a","b","c","d";1,2,3,4}) equals 3

LOOKUP("bump",{"a",1;"b",2;"c",3}) equals 2

#### Related Functions

| | |
|---|---|
| HLOOKUP | Looks in the first row of an array and moves down the column to return the value of a cell |
| INDEX | Uses an index to choose a value from a reference or array |
| VLOOKUP | Looks in the first column of an array and moves across the row to return the value of a cell |

## LOWER(text)

Converts all uppercase letters in a text string to lowercase.

***Text*** is the text you want to convert to lowercase. LOWER does not change characters in *text* that are not letters.

#### Examples

LOWER("E. E. Cummings") equals "e. e. cummings"

LOWER("Apt. 2B") equals "apt. 2b"

Lower is similar to PROPER and UPPER. For more examples, see PROPER.

#### Related Functions

| | |
|---|---|
| PROPER | Capitalizes the first letter in each word of a text value |
| UPPER | Capitalizes every letter in a text value |

*Macro Sheets Only*

## MAIN.CHART(type_num,stack,100,vary,overlap,drop,hilo,overlap%,cluster,angle)

## MAIN.CHART?(type_num,stack,100,vary,overlap,drop,hilo,overlap%,cluster,angle)

Equivalent to the Format Main Chart command in Microsoft Excel version 2.2 or earlier. This function is included only for macro compatibility. To format main charts in Microsoft Excel version 3.0, use the FORMAT.MAIN function, which is equivalent to the Format Main Chart command.

*Macro Sheets Only*

## MAIN.CHART.TYPE(type_num)

Equivalent to the Chart Main Chart Type command in Microsoft Excel for the Macintosh version 1.5 or earlier. This function is included only for macro compatibility. To format a main chart or to specify a main chart type in Microsoft Excel version 3.0, use the FORMAT.MAIN function, which is equivalent to the Format Main Chart command.

## MATCH(lookup_value,lookup_array, match_type)

Returns the relative position of an element in an array that matches a specified value in a specified way. Use MATCH instead of one of the lookup functions when you need the position of a matched item instead of the item itself.

***Lookup_value*** is the value you use to find the value you want in a table.

- *Lookup_value* is the value you want to match in *lookup_array*. For example, when you look up someone's number in a telephone book, you are using the person's name as the lookup value, but the telephone number is the value you want.
- *Lookup_value* can be a value (number, text, or logical value) or a cell reference to a number, text, or logical value.

***Lookup_array*** is a contiguous range of cells containing possible lookup values. *Lookup_array* can be an array or an array reference.

***Match_type*** is the number –1, 0, or 1. *Match_type* specifies how Microsoft Excel matches *lookup_value* with values in *lookup_array*.

- If *match_type* is 1, MATCH finds the largest value that is less than or equal to *lookup_value*. *Lookup_array* must be placed in ascending order: ...–2, –1, 0, 1, 2,...A–Z, FALSE, TRUE, and so on.
- If *match_type* is 0, MATCH finds the first value that is exactly equal to *lookup_value*. *Lookup_array* can be in any order.
- If *match_type* is –1, MATCH finds the smallest value that is greater than or equal to *lookup_value*. *Lookup_array* must be placed in descending order: TRUE, FALSE, Z–A,...2, 1, 0, –1, –2,..., and so on.
- If *match_type* is omitted, it is assumed to be 1.

### Notes

- MATCH returns the position of the matched value within *lookup_array*, not the value itself. For example:
  MATCH("b",{"a","b","c"},0) returns 2, the relative position of "b" within the array {"a","b","c"}.
- MATCH does not distinguish between uppercase and lowercase letters when matching text values.
- If MATCH is unsuccessful in finding a match, it returns the #N/A error value.
- If *match_type* is 0 and *lookup_value* is text, *lookup_value* can contain the wildcard characters, asterisk (*) and question mark (?). An asterisk matches any sequence of characters; a question mark matches any single character.

### Examples

| | A | B | C |
|---|---|---|---|
| 1 | Income (in Yen) | U.S. Dollars | U.S. Tax Rate |
| 2 | ¥5,365,000.00 | $37,000.00 | 21.50% |
| 3 | ¥5,510,000.00 | $38,000.00 | 21.67% |
| 4 | ¥5,655,000.00 | $39,000.00 | 21.84% |
| 5 | ¥5,800,000.00 | $40,000.00 | 21.99% |
| 6 | ¥5,945,000.00 | $41,000.00 | 22.14% |
| 7 | ¥6,090,000.00 | $42,000.00 | 22.28% |
| 8 | ¥6,235,000.00 | $43,000.00 | 22.41% |
| 9 | ¥6,380,000.00 | $44,000.00 | 22.54% |
| 10 | ¥6,525,000.00 | $45,000.00 | 22.67% |
| 11 | ¥6,670,000.00 | $46,000.00 | 22.90% |
| 12 | ¥6,815,000.00 | $47,000.00 | 23.11% |
| 13 | ¥6,960,000.00 | $48,000.00 | 23.32% |
| 14 | ¥7,105,000.00 | $49,000.00 | 23.51% |
| 15 | ¥7,249,998.55 | $49,999.99 | 23.69% |

In the preceding worksheet:

MATCH(39000,B2:B15,1) equals 3

MATCH(49800,B2:B15) equals 13

MATCH(38000,B2:B15,0) equals 2

MATCH(39000,B2:B15,–1) equals #N/A because the range B2:B15 is ordered incorrectly for type –1 matching (the order must be descending to be correct).

MATCH("22*",C2:C15,0) equals 5 because MATCH looks for the first text beginning with 22. (Note that C2:C15 contains text formatted as percent numbers.)

MATCH("2?.32%",C2:C15,0) equals 12

Suppose Yen refers to A2:A15, YenDollar to A2:C15, and MyIncome to a cell containing the number 6,301,126.33¥. This formula:

"Your tax rate is "&LOOKUP(MyIncome,YenDollar)&", which places you in tax bracket number "&MATCH(MyIncome,Yen)&"."



produces this result:

"Your tax rate is 22.41%, which places you in tax bracket number 7."

### Related Functions

| | |
|---|---|
| HLOOKUP, VLOOKUP, and LOOKUP | Look up values in a reference or array |
| INDEX | Uses an index to choose a value from a reference or an array |

## MAX(number1,number2,...)

Returns the largest number in the list of arguments.

***Number1, number2,...*** are 1 to 14 numbers for which you want to find the maximum value.

- You can specify arguments that are numbers, empty cells, logical values, or text representations of numbers. Arguments that are error values or text that cannot be translated into numbers cause errors.
- If an argument is an array or reference, only numbers in that array or reference are used. Empty cells, logical values, text, or error values in the array or reference are ignored.
- If the arguments contain no numbers, MAX returns 0.

### Examples

If A1:A5 contains the numbers 10, 7, 9, 27, and 2, then:

MAX(A1:A5) equals 27

MAX(A1:A5,30) equals 30

Suppose you have a worksheet containing columns of text and one column of numbers (each row contains several text cells and one number cell). The following macro finds the row on the active worksheet containing the largest number and selects this row so that it will be easier to find among the other rows:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | FindMaximumValue | |
| 4 | | =SELECT(!$A$1) | Searches from first cell |
| 5 | MaxValue | =MAX(!$1:$16384) | Searches entire worksheet |
| 6 | | =FORMULA.FIND(MaxValue,2,1,1,1) | Moves to cell with max value |
| 7 | | =SELECT("R") | Selects that row |
| 8 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| DMAX | Returns the maximum value from selected database entries |
| MIN | Returns the minimum value in a list of arguments |

## MDETERM(array)

Returns the matrix determinant of an array.

***Array*** is a numeric array with an equal number of rows and columns.

- *Array* can be given as a cell range, for example, A1:C3; as an array constant, such as {1,2,3;4,5,6;7,8,9}; or as a name to either of these.
- If any cells in *array* are empty or contain text, MDETERM returns the #VALUE! error value.
- MDETERM also returns #VALUE! if *array* does not have an equal number of rows and columns.

### Notes

- The matrix determinant is a number derived from the values in *array*. For a three-row, three-column array, A1:C3, the determinant is defined as:

  MDETERM(A1:C3) equals
  A1×(B2×C3–B3×C2) + A2×(B3×C1–B1×C3) + A3×(B1×C2–B2×C1)
- Matrix determinants are generally used for solving systems of mathematical equations that involve several variables.

- MDETERM is calculated with an accuracy of approximately 16 digits, which may lead to a small numerical error when the cancellation is not complete. For example, the determinant of a singular matrix may differ from zero by 1E–16. For more information on arrays, see Chapter 7, "Analyzing and Calculating a Worksheet," in the *Microsoft Excel User's Guide*.

### Examples

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | 1 | 3 | 8 | 5 | |
| 2 | 1 | 3 | 6 | 1 | |
| 3 | 1 | 1 | 1 | 0 | |
| 4 | 7 | 9 | 10 | 2 | |

In the preceding worksheet:

MDETERM(A1:D4) equals –20

MDETERM(B2:D4) equals –5

MDETERM(B2:C3) equals –3

MDETERM(A1:D2) equals #VALUE! because the range A1:D2 does not have an equal number of rows and columns.

MDETERM(A1:E5) equals #VALUE! because A1:E5 contains empty cells.

### Related Functions

| | |
|---|---|
| MINVERSE | Returns the matrix inverse of an array |
| MMULT | Returns the matrix product of two arrays |
| TRANSPOSE | Returns the transpose of an array |

## MEDIAN(number1,number2,...)

Returns the median of the given numbers. The median is the number in the middle of a set of numbers; that is, half the numbers occur above the median and half below.

***Number1, number2,...*** are 1 to 14 numbers for which you want the median.

- The arguments should be numbers, or names, arrays, or references that contain numbers.
- If an array or reference argument contains text, logical values, or empty cells, those values are ignored; however, cells with the value zero are included.

### Note

If there is an even number of numbers in the set, then MEDIAN calculates the average of the two numbers in the middle. See the second example below.

### Examples

MEDIAN(1,2,3,4,5) equals 3

MEDIAN(1,2,3,4,5,6) equals 3.5, the average of 3 and 4

### Related Functions

| | |
|---|---|
| AVERAGE | Returns the average of its arguments |
| COUNT | Counts how many numbers are in the list of arguments |
| COUNTA | Counts how many values are in the list of arguments |
| DAVERAGE | Returns the average of selected database entries |
| SUM | Adds its arguments |

*Macro Sheets Only*

## MERGE.STYLES(document_text)

Equivalent to choosing the Merge button in the Format Style dialog box. Merges all the styles from another document into the active document. Use MERGE.STYLES when you want to import styles from another document. For detailed information on the Format Style command, see online Help.

***Document_text*** is the name of a document from which you want to merge styles into the active document.

### Note

If any styles from the document being merged have the same name as styles in the active document, a dialog box appears asking if you want to replace the existing definitions of the styles with the "merged" definitions of the styles. If you choose Yes, all the definitions are replaced; if you choose No, all the original definitions in the active document are retained.



### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula merges the styles from the document REPORT.XLS into the active document:

MERGE.STYLES("REPORT.XLS")

In Microsoft Excel for the Macintosh, the following macro formula merges the styles from the document Report into the active document:

MERGE.STYLES("Report")

### Related Functions

| | |
|---|---|
| DEFINE.STYLE | Creates or changes a cell style |
| DELETE.STYLE | Deletes a cell style |

*Macro Sheets Only*

## MESSAGE(logical,text)

Displays and removes messages in the message area of the status bar. (For information on the status bar, see Chapter 1, "Microsoft Excel Basics," in the *Microsoft Excel User's Guide*.) MESSAGE is useful for displaying text that doesn't need a response, such as commands in user-defined menus.

***Logical*** is a logical value specifying whether to display or remove a message.

- If *logical* is TRUE, Microsoft Excel displays *text* in the message area of the status bar.
- If *logical* is FALSE, Microsoft Excel removes any messages, and the status bar is returned to normal (that is, command help messages are displayed).

***Text*** is the message you want to display in the status bar. If *text* is "" (empty text), Microsoft Excel removes any messages currently displayed in the status bar.

### Notes

- Only one message can be displayed in the status bar at a time. Messages are always displayed in the same place.
- MESSAGE works the same way whether the status bar is displayed or not. You can, for example, use MESSAGE while the status bar isn't displayed. As soon as you display the status bar, you see your message.
- If you display any message (even empty text) and don't remove it with MESSAGE(FALSE), that message is displayed until you quit Microsoft Excel.

### Example

The following lines in a macro display a message warning that you must wait for a moment while the macro calls a subroutine. Only the last few lines of the calling macro are shown. Assume that the subroutine FindRoot is defined on another part of the worksheet.

| | A | B | C |
|---|---|---|---|
| 33 | | =BEEP() | Signals the message |
| 34 | | =MESSAGE(TRUE,"One moment please...") | Displays the message |
| 35 | | =FindRoot(Polynomial) | Calls the subroutine |
| 36 | | =MESSAGE(FALSE) | |
| 37 | | =RETURN() | |

You could also use the ALERT function to get the user's attention; however, this would interrupt the macro and would require the user's intervention before the macro could continue.

### Related Functions

| | |
|---|---|
| ALERT | Displays a dialog box with a message |
| BEEP | Sounds a tone |

## MID(text,start_num,num_chars)

Returns a specific number of characters from a text string, starting at the position you specify.

***Text*** is the text string containing the characters you want to extract.

***Start_num*** is the position of the first character you want to extract in *text*. The first character in *text* has *start_num* 1, and so on.

- If *start_num* is greater than the length of *text*, MID returns "" (empty text).
- If *start_num* is less than the length of *text*, but *start_num* plus *num_chars* exceeds the length of *text*, MID returns the characters up to the end of *text*.
- If *start_num* is less than 1, MID returns the #VALUE! error value.

***Num_chars*** specifies how many characters to return from text. If *num_chars* is negative, MID returns the #VALUE! error value.

### Examples

MID("Fluid Flow",1,5) equals "Fluid"

MID("Fluid Flow",7,20) equals "Flow"

MID("1234",5,5) equals "" (empty text)

For more examples, see CODE and FIND.

### Related Functions

| | |
|---|---|
| FIND | Finds one text value within another (case sensitive) |
| LEFT | Extracts the leftmost characters from a text value |
| RIGHT | Extract the rightmost characters from a text value |
| SEARCH | Finds one text value within another (not case sensitive) |

## MIN(number1,number2,...)

Returns the smallest number in the list of arguments.

***Number1, number2,...*** are 1 to 14 numbers for which you want to find the minimum value.

- You can specify arguments that are numbers, empty cells, logical values, or text representations of numbers. Arguments that are error values or text that cannot be translated into numbers cause errors.
- If an argument is an array or reference, only numbers in that array or reference are used. Empty cells, logical values, text, or error values in the array or reference are ignored.
- If the arguments contain no numbers, MIN returns 0.

### Examples

If A1:A5 contains the numbers 10, 7, 9, 27, and 2, then:

MIN(A1:A5) equals 2

MIN(A1:A5,0) equals 0

MIN is similar to MAX; for more examples, see MAX.

### Related Functions

| | |
|---|---|
| DMIN | Returns the minimum value from selected database entries |
| MAX | Returns the maximum value in a list of arguments |

## MINUTE(serial_number)

Returns the minute corresponding to *serial_number*. The minute is given as an integer, ranging from 0 to 59.

***Serial_number*** is the date-time code used by Microsoft Excel for date and time calculations. You can give *serial_number* as text, such as "16:48:00" or "4:48:00 P.M.", instead of as a number. The text is automatically converted to a serial number. For more information about *serial_number*, see NOW.

### Note

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.



### Examples

MINUTE("4:48:00 PM") equals 48

MINUTE(0.01) equals 14

MINUTE(4.02) equals 28

### Related Functions

| | |
|---|---|
| NOW | Returns the serial number of the current date and time |
| YEAR, MONTH, DAY, WEEKDAY, HOUR, and SECOND | Convert serial numbers into years, months, days, days of the week, hours, or seconds |

## MINVERSE(array)

Returns the inverse matrix for the matrix stored in an array.

***Array*** is a numeric array with an equal number of rows and columns.

- *Array* can be given as a cell range, for example, A1:C3; as an array constant, such as {1,2,3;4,5,6;7,8,9}; or as a name for either of these.
- If any cells in *array* are empty or contain text, MINVERSE returns the #VALUE! error value.
- MINVERSE also returns #VALUE! if *array* does not have an equal number of rows and columns.

### Notes

- Formulas that return arrays must be entered as array formulas. For information on arrays, see Chapter 7, "Analyzing and Calculating a Worksheet," in the *Microsoft Excel User's Guide*.
- Inverse matrices, like determinants, are generally used for solving systems of mathematical equations involving several variables. The product of a matrix and its inverse is the identity matrix—the square array in which the diagonal values equal 1 and all other values equal 0. For example, the following illustration shows the two-row, two-column identity matrix in cells A1:B2, and the four-row, four-column identity matrix in cells D1:G4:

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | | 1 | 0 | 0 | 0 |
| 2 | 0 | 1 | | 0 | 1 | 0 | 0 |
| 3 | | | | 0 | 0 | 1 | 0 |
| 4 | | | | 0 | 0 | 0 | 1 |
| 5 | | | | | | | |

- The following worksheet shows how the inverse of a two-row, two-column matrix is calculated. The letters a, b, c, and d represent any four numbers. The matrix D1:E2 is the inverse of the matrix A1:B2 on this worksheet:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | a | b | | d/(a*d–b*c) | b/(b*c–a*d) |
| 2 | c | d | | c/(b*c–a*d) | a/(a*d–b*c) |
| 3 | | | | | |

- MINVERSE is calculated with an accuracy of approximately 16 digits, which may lead to a small numerical error when the cancellation is not complete.
- Some square matrices cannot be inverted, and will return the #NUM! error value with MINVERSE. The determinant for a noninvertable matrix is 0.

### Examples

| | A | B | C | D |
|---|---|---|---|---|
| 1 | 1 | 2 | 1 | |
| 2 | 3 | 4 | –1 | |
| 3 | 0 | 2 | 0 | |
| 4 | | | | |

In the worksheet above:

MINVERSE(B2:C3) equals {0,0.5;–1,2}

MINVERSE(A1:C3) equals {0.25,0.25,–0.75;0,0,0.5; 0.75,–0.25,–0.25}

MINVERSE(A1:D4) equals #VALUE!, because A1:D4 contains empty cells.

### Tip

Use the INDEX function to access individual elements from the inverse matrix. For example, in the worksheet above:

INDEX(MINVERSE(A1:C3),1,1) equals 0.25

INDEX(MINVERSE(B2:C3),2,2) equals 2

### Related Functions

| | |
|---|---|
| MDETERM | Returns the matrix determinant of an array |
| MMULT | Returns the matrix product of two arrays |
| TRANSPOSE | Returns the transpose of an array |

## MIRR(values,finance_rate,reinvest_rate)

Returns the modified internal rate of return for a series of periodic cash flows. MIRR considers both the cost of the investment and the interest received on reinvestment of cash.

***Values*** is an array or a reference to cells that contain numbers. These numbers represent a series of payments (negative values) and income (positive values) occurring at regular periods.

- *Values* must contain at least one positive value and one negative value to calculate the modified internal rate of return. Otherwise, MIRR returns the #DIV/0! error value.
- Text, logical, and empty cell *values* are ignored.

***Finance_rate*** is the interest rate you pay on the money used in the cash flows.

***Reinvest_rate*** is the interest rate you receive on the cash flows as you reinvest them.

### Notes

- MIRR uses the order of *values* to interpret the order of cash flows. Be sure to enter your payment and income values in the sequence you want and with the correct signs (positive values for cash received, negative values for cash paid).
- If *n* is the number of cash flows in *values*, frate is the *finance_rate*, and rrate is the *reinvest_rate*, then the formula for MIRR is:

$$\left(\frac{-\text{NPV}(rrate,values[positive]) * (1+rrate)^{n}}{\text{NPV}(frate,values[negative]) * (1+frate)}\right)^{\frac{1}{n-1}} - 1$$

### Examples

Suppose you're a commercial fisherman just completing your fifth year of operation. Five years ago, you borrowed $120,000 at 10% annual interest to purchase a boat. Your catches have yielded $39,000, $30,000, $21,000, $37,000, and $46,000. During these years you reinvested your profits, earning 12% annually.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | Initial Cost | ($120,000) | | |
| 2 | Year 1 Income | $39,000 | | |
| 3 | Year 2 Income | $30,000 | | |
| 4 | Year 3 Income | $21,000 | | |
| 5 | Year 4 Income | $37,000 | | |
| 6 | Year 5 Income | $46,000 | | |

To calculate the investment's modified rate of return after five years:

MIRR(B1:B6,10%,12%) equals 12.61%

To calculate the modified rate of return after three years:

MIRR(B1:B4,10%,12%) equals –4.80%

To calculate the five-year modified rate of return based on a *reinvest_rate* of 14%:

MIRR(B1:B6,10%,14%) equals 13.48%

### Related Functions

| | |
|---|---|
| IRR | Returns the internal rate of return for an investment, without financing costs or reinvestment gains |
| RATE | Returns the interest rate per period of an investment |



## MMULT(array1,array2)

Returns the matrix product of two arrays. The result is an array with the same number of rows as *array1* and the same number of columns as *array2*.

***Array1,array2*** are the arrays you want to multiply.

- The number of columns in *array1* must be the same as the number of rows in *array2*, and both arrays must contain only numbers.
- *Array1* and *array2* can be given as cell ranges, array constants, or references.
- If any cells are empty or contain text, or if the number of columns in *array1* is different from the number of rows in *array2*, MMULT returns the #VALUE! error value.

### Notes

- The matrix product array *a* of two arrays *b* and *c* is:

$$a_{ij} = \sum_{k=1}^{n} b_{ik} c_{kj}$$

where *i* is the row number and *j* is the column number.

- Formulas that return arrays must be entered as array formulas. For information on arrays, see Chapter 7, "Analyzing and Calculating a Worksheet," in the *Microsoft Excel User's Guide*.

### Examples

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | 1 | 3 | 0 | 2 | 0 |
| 2 | 7 | 2 | 0 | 0 | 2 |
| 3 | 1 | 0 | 0 | | |
| 4 | | | | | |

In the worksheet above:

MMULT(A1:B2,D1:E2) equals {2,6;14,4}

MMULT(B1:C2,D1:E2) equals {6,0;4,0}

MMULT(A1:C3,D1:E2) equals #VALUE!, because A1:C3 has three columns and D1:E2 has only two rows.

### Related Functions

| | |
|---|---|
| MDETERM | Returns the matrix determinant of an array |
| MINVERSE | Returns the matrix inverse of an array |
| TRANSPOSE | Returns the transpose of an array |

## MOD(number,divisor)

Returns the remainder (modulus) after *number* is divided by *divisor*. The result has the same sign as *divisor*.

***Number*** is the number for which you want to find the remainder.

***Divisor*** is the number by which you want to divide *number*. If *divisor* is 0, MOD returns the #DIV/0! error value.

### Note

The MOD function can be expressed in terms of the INT function:
MOD(*n*,*d*) = *n* – *d**INT(*n*/*d*)

### Examples

MOD(3,2) equals 1

MOD(–3,2) equals 1

MOD(3,–2) equals –1

MOD(–3,–2) equals –1

### Related Functions

| | |
|---|---|
| INT | Rounds a number down to the nearest integer |
| ROUND | Rounds a number to a specified number of digits |
| TRUNC | Truncates a number to an integer |

## MONTH(serial_number)

Returns the month corresponding to *serial_number*. The month is given as an integer, ranging from 1 (January) to 12 (December).

***Serial_number*** is the date-time code used by Microsoft Excel for date and time calculations. You can give *serial_number* as text, such as "4-15-1991" or "15-Apr-1991", instead of as a number. The text is automatically converted to a serial number. For more information about *serial_number*, see NOW.

### Note

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

### Tip

To display the result of the MONTH function as a month name, use the Format Number command to format the cell containing the function with the custom number format "mmmm".

### Examples

MONTH("6-May") equals 5

MONTH(366) equals 12

MONTH(367) equals 1

TEXT(MONTH(367),"mmmm") equals January

### Related Functions

| | |
|---|---|
| YEAR, DAY, WEEKDAY, HOUR, MINUTE, and SECOND | Convert serial numbers into years, days, days of the week, hours, minutes, or seconds |

*Macro Sheets Only*

## MOVE(x_pos,y_pos,window_text)
## MOVE?(x_pos,y_pos,window_text)

Moves a window so that its upper-left corner is at the specified horizontal and vertical positions. Using MOVE in a macro is equivalent to moving a window by dragging its title bar. MOVE is also equivalent to the Move command on the Control menu in Microsoft Windows or OS/2 Presentation Manager. The dialog-box form, MOVE?, is used only in Microsoft Excel for Windows or OS/2, although it is supported for macro compatibility in Microsoft Excel for the Macintosh.

***X_pos*** is the horizontal position to which you want to move the window. *X_pos* is measured in points.

- In Microsoft Excel for Windows or OS/2, *x_pos* is measured from the left edge of your workspace to the left edge of the window.
- In Microsoft Excel for the Macintosh, *x_pos* is measured from the left edge of your screen to the left edge of the window.

***Y_pos*** is the vertical position where you want to move the window. *Y_pos* in measured in points.

- In Microsoft Excel for Windows or OS/2, *y_pos* is measured from the top edge of your workspace to the top edge of the window.
- In Microsoft Excel for the Macintosh, *y_pos* is measured from the bottom edge of the formula bar to the top edge of the window.

***Window_text*** is the name of the window you want to move. If *window_text* is not specified, it is assumed to be the active window.

### Notes

- MOVE does not change the size of the window or affect whether the specified window is active or inactive.
- In Microsoft Excel for OS/2 and Microsoft Excel for the Macintosh, if *window_text* is "Clipboard", MOVE moves the Clipboard. The Clipboard must already be available; if it is not available, choose the Window Show Clipboard command or use the SHOW.CLIPBOARD function before using the MOVE function.
- Most macro functions that describe movement or position on the screen take arguments measured in points. One point is equal to 1/72nd of an inch (72 points are equal to 1 inch). Points are commonly used in typesetting—the height of fonts in Microsoft Excel, for example, is measured in points.

### Example

In Microsoft Excel for Windows or OS/2, the following macro formula places the active window against the upper-left edge of the Microsoft Excel workspace:

MOVE(1,1)

In Microsoft Excel for the Macintosh, the following macro formula places the active window against the upper-left edge of the screen:

MOVE(1,1)

### Related Functions

| | |
|---|---|
| FORMAT.MOVE | Moves the selected object |
| SIZE | Changes the size of a window |

## N(value)

Returns a value converted to a number.

***Value*** is the value you want converted. N converts values as follows:

| If *value* is or refers to | N returns |
|---|---|
| A number | That number |
| A date, in one of Microsoft Excel's built-in date formats | The serial number of that date |
| TRUE | 1 |
| Anything else | 0 |



### Note

It is not generally necessary to use the N function in a formula, since Microsoft Excel automatically converts values as necessary. This function is provided for compatibility with other spreadsheet programs. For information on how Microsoft Excel converts values, see "Converting Data Types" earlier in this book. Also see Chapter 19, "Designing and Writing a Command Macro," in the *Microsoft Excel User's Guide*.

### Examples

| | A | B | C | D |
|---|---|---|---|---|
| 1 | 7 | odd | TRUE | |
| 2 | 8 | even | TRUE | |
| 3 | 13 | odd | FALSE | |
| 4 | 14 | even | FALSE | |

In the preceding worksheet:

N(A1) equals 7

N(B2) equals 0, because B2 contains text

N(C2) equals 1, because C2 contains TRUE

N("7") equals 0, because "7" is text

N("4/17/91") equals 0, because "4/17/91" is text

### Related Functions

| | |
|---|---|
| GET.CELL | Returns information about the formatting, location, or contents of a cell |
| T | Converts its argument into text |

## NA()

Returns the error value #N/A. #N/A is the error value that means "no value is available." Use NA to mark empty cells. By entering #N/A in cells where you are missing information, you can avoid the problem of unintentionally including empty cells in your calculations. (When a formula refers to a cell containing #N/A, the formula returns the error value #N/A.)

### Notes

- You must include the empty parentheses with the function name. Otherwise, Microsoft Excel will not recognize it as a function.
- You can also type the value #N/A directly into a cell. The NA function is provided for compatibility with other spreadsheet programs.

### Related Function

| | |
|---|---|
| ISNA | Returns TRUE if a value is the error value #NA |

Macro Sheets Only

## NAMES(*document_text,type_num*)

Returns a horizontal text array of all the names defined in a document. The returned array lists the names in alphabetical order. Use NAMES instead of LIST.NAMES when you want to return the active worksheet's names to the macro sheet instead of to the active worksheet.

***Document_text*** is the name of the document whose names you want to find. If *document_text* is omitted, it is assumed to be the active document.

***Type_num*** is a number from 1 to 3 that specifies whether to include hidden names in the returned array.

| If *type_num* is | NAMES returns |
|---|---|
| 1 or omitted | Normal names only |
| 2 | Hidden names only |
| 3 | All names |

### Notes

- Hidden names are defined using the DEFINE.NAME macro function and do not appear in the Formula Paste Name, Formula Define Name, or Formula Goto dialog boxes. If you use macros to develop custom applications in Microsoft Excel, use hidden names to avoid cluttering name lists in dialog boxes with confusing or unfamiliar names. For more information, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.
- NAMES returns a horizontal array, so you will normally enter this function as an array in several horizontal cells or define a name to refer to the array that NAMES returns. If you want the names in a vertical array instead, use the TRANSPOSE function. See the example below.
- You can also use the COLUMNS functions to count the number of entries in the array.

### Examples

The following macro formula places the names defined on the active worksheet in a vertical array named VerticalNameArray:

```
SET.NAME("VerticalNameArray",TRANSPOSE(NAMES()))
```

Suppose you have a worksheet with a group of names defined as absolute references to cells, and you need to copy those names from one worksheet to another. You can use the NAMES function to return an array of names defined on the first worksheet, the INDEX function to retrieve each name, and the GET.NAME and DEFINE.NAME functions to apply the names and their definitions to the second worksheet, as shown in the following macro:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | CopyNameDefinitions | |
| 4 | FirstSheet | =INPUT("Enter the first worksheet:",2) | |
| 5 | SecondSheet | =INPUT("Enter the second worksheet:",2) | |
| 6 | | =ECHO(FALSE) | |
| 7 | | =ACTIVATE(FirstSheet) | Switches to FirstSheet |
| 8 | | =SET.NAME("NameArray",NAMES()) | Enters names in array |
| 9 | NumNames | =COLUMNS(NameArray) | Number of names? |
| 10 | | =FOR("Counter",1,NumNames) | Loops once for each name |
| 11 | NextName | =INDEX(NameArray,1,Counter) | Retrieves the next name |
| 12 | | =ACTIVATE(FirstSheet) | |
| 13 | NextDef | =GET.NAME("r"&NextName) | Gets definition of next name |
| 14 | | =ACTIVATE(SecondSheet) | |
| 15 | | =DEFINE.NAME(NextName,DEREF(NextDef | Defines the name locally |
| 16 | | =NEXT() | |
| 17 | | =RETURN() | |



The complete formula in cell B15 is:

```
=DEFINE.NAME(NextName,DEREF(NextDef))
```

### Related Functions

| | |
|---|---|
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| DELETE.NAME | Deletes a name |
| GET.DEF | Returns a name matching a definition |
| GET.NAME | Returns the definition of a name |
| LIST.NAMES | Lists names in a worksheet or macro sheet along with information about the names |
| SET.NAME | Defines a name on a macro sheet |

Macro Sheets Only

## NEW(*type_num, xy_series*)
## NEW?(*type_num,xy_series*)

Equivalent to the File New command. Creates a new Microsoft Excel document or opens a template. For detailed information on the File New command, see online Help.

***Type_num*** specifies the type of file to create, as follows:

| *Type_num* | Document |
|---|---|
| 1 | Worksheet |
| 2 | Chart |
| 3 | Macro sheet |
| 4 | International macro sheet |
| Quoted text | Template |

### Note

You can only use NEW to create new sheets from templates that exist in the startup directory or folder, using for *type_num* the text that appears in the File New list box. To create new sheets from any template, use the OPEN function. For more information on templates, see Chapter 16, "Customizing the Workspace and Protecting Data," in the *Microsoft Excel User's Guide*.

***Xy_series*** is a number from 0 to 3 that specifies how data is arranged in a chart.

| *Xy_series* | Result |
|---|---|
| 0 | Displays dialog box if selection is ambiguous. |
| 1 or omitted | First row/column is the first data series. |
| 2 | First row/column are the category (x) axis labels. |
| 3 | First row/column are x-values; created chart is a scatter chart. |

### Example

The following macro formula creates a new chart with the data in the first row/column as data values/y-axis values:

```
NEW(2,2)
```

### Related Functions

| | |
|---|---|
| NEW.WINDOW | Creates a new window for an existing worksheet or macro sheet |
| OPEN | Opens an existing file or workspace |

Macro Sheets Only

## NEW.WINDOW()

Equivalent to the Window New Window command. Creates a new window for the active worksheet or macro sheet. An error is returned if a chart, or a window that is part of a workgroup, is the active document. For detailed information on the Window New Window command, see online Help.

After you use NEW.WINDOW, use the MOVE, SIZE, and ARRANGE.ALL functions to size and position the new window.

### Related Functions

| | |
|---|---|
| ARRANGE.ALL | Arranges all displayed windows to fill the workspace |
| MOVE | Moves a window |
| SIZE | Changes the size of a window |

Macro Sheets Only

## NEXT()

Ends a FOR–NEXT, FOR.CELL–NEXT, or WHILE–NEXT loop and continues carrying out the current macro with the formula that follows the NEXT function. For information on loops, see FOR, WHILE, or FOR.CELL in this book, or Chapter 19, "Designing and Writing a Command Macro," in the *Microsoft Excel User's Guide*.

Macro Sheets Only

## NOT(logical)

Reverses the value of its argument. Use NOT when you want to make sure a value is not equal to one particular value.

***Logical*** is a value or expression that can be evaluated to TRUE or FALSE. If *logical* is FALSE, NOT returns TRUE; if *logical* is TRUE, NOT returns FALSE.

### Examples

NOT(FALSE) equals TRUE

NOT(1+1=2) equals FALSE

### Related Functions

| | |
|---|---|
| AND | Returns TRUE if all its arguments are TRUE |
| OR | Returns TRUE if any argument is TRUE |

Macro Sheets Only

## NOTE(add_text,cell_ref,start_char,num_chars)
## NOTE?()

Equivalent to the Formula Note command. Creates a note or replaces characters in a note. Use NOTE to create a cell note or to replace a specified number of characters with other text.

***Add_text*** is text of up to 255 characters you want to add to a note. *Add_text* must be enclosed in quotation marks. If *add_text* is omitted, it is assumed to be "" (empty text).

***Cell_ref*** is the cell to which you want to add the note text. If *cell_ref* is omitted, *add_text* is added to the active cell's note.

***Start_char*** is the number of the character at which you want *add_text* to be added. If *start_char* is omitted, it is assumed to be 1.

***Num_chars*** is the number of characters that you want to replace in the note. If *num_chars* is omitted, it is assumed to be equal to the length of the note.

### Notes

- NOTE returns the number of the last character entered in the cell note. This is useful if you want to know how many characters are in the text string so it can be manipulated with other functions such as RIGHT, LEFT, and MID.
- The dialog-box form of this function, NOTE?, takes no arguments.

### Examples

NOTE() deletes the note attached to the active cell.

The following macro formula attaches a new note, "Attached are the first fiscal quarter sales figures", to cell A1 on the active document:

```
NOTE("Attached are the first fiscal quarter sales figures",!$A$1)
```

The following macro formula inserts the text, "Assumptions begin with this row", at position 1000 in the note attached to cell A1, or at the end of the note if the note has fewer than 1,000 characters:

```
NOTE("Assumptions begin with this row",!$A$1,1000)
```



### Related Function

| | |
|---|---|
| GET.NOTE | Returns characters from a note |

## NOW()

Returns the serial number of the current date and time.

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different default date systems. Microsoft Excel for Windows or OS/2 uses the 1900 Date System, in which serial numbers range from 1 to 65,380, corresponding to the dates January 1, 1900 through December 31, 2078. Microsoft Excel for the Macintosh uses the 1904 Date System, in which serial numbers range from 0 to 63,918, corresponding to the dates January 1, 1904 through December 31, 2078.

Numbers to the right of the decimal point in the serial number represent the time; numbers to the left represent the date. For example, in the 1900 Date System, the serial number 367.5 represents the date time combination 12:00 P.M., January 1, 1901.

You can change the date system by turning on or off the 1904 Date System check box in the Options Calculation dialog box.

The date system is changed automatically when you open a document from another platform. For example, if you are working in Microsoft Excel for Windows and you open a document created in Microsoft Excel for the Macintosh, the 1904 Date System check box is turned on automatically.

### Note

The NOW function changes only when the worksheet is calculated or when the macro containing the function is run. It is not updated continuously.

### Examples

If you are using the 1900 Date System and your computer's built-in clock is set to 12:30:00 P.M., 1-Jan-1987, then:

NOW() equals 31778.52083

Ten minutes later:

NOW() equals 31778.52778

### Related Functions

| | |
|---|---|
| TODAY | Returns the serial number of today's date |
| YEAR, MONTH, DAY, WEEKDAY, HOUR, MINUTE, and SECOND | Convert serial numbers to years, months, days, days of the week, hours, minutes, and seconds |

## NPER(rate,pmt,pv,fv,type)

Returns the number of periods for an investment based on periodic, constant payments and a constant interest rate.

For a more complete description of the arguments in NPER and for more information on annuity functions, see PV.

***Rate*** is the interest rate per period.

***Pmt*** is the payment made each period; it cannot change over the life of the annuity. Typically, *pmt* contains principal and interest but no other fees or taxes.

***Pv*** is the present value, or the lump-sum amount that a series of future payments is worth right now.

***Fv*** is the future value, or a cash balance you want to attain after the last payment is made. If *fv* is omitted, it is assumed to be 0 (the future value of a loan, for example, is 0).

***Type*** is the number 0 or 1 and indicates when payments are due. If *type* is omitted, it is assumed to be 0.

| Set *type* equal to | If payments are due |
|---|---|
| 0 | At the end of the period |
| 1 | At the beginning of the period |

### Examples

NPER(12%/12,–100,–1000,10000,1) equals 60

NPER(1%,–100,–1000,10000) equals 60

NPER(1%,–100,1000) equals 11

### Related Functions

| | |
|---|---|
| FV | Returns the future value of an investment |
| IPMT | Returns the interest payment for an investment for a given period |
| PMT | Returns the periodic total payment for an investment |
| PPMT | Returns the payment on the principal for an investment for a given period |
| PV | Returns the present value of an investment |
| RATE | Returns the interest rate per period of an investment |

## NPV(*rate,value1,value2,...*)

Returns the net present value of an investment based on a series of periodic cash flows and a discount rate. The net present value of an investment is today's value of a series of future payments (negative values) and income (positive values).

***Rate*** is the rate of discount over the length of one period.

***Value1, value2,...*** are 1 to 13 arguments representing the payments and income.

- *Value1*, *value2*,... must be equally spaced in time and occur at the end of each period.
- NPV uses the order of *value1*,*value2*,... to interpret the order of cash flows. Be sure to enter your payment and income values in the correct sequence.
- Arguments that are numbers, empty cells, logical values, or text representations of numbers are counted; arguments that are error values or text that cannot be translated into numbers are ignored.
- If an argument is an array or reference, only numbers in that array or reference are counted. Empty cells, logical values, text, or error values in the array or reference are ignored.

### Notes

- The NPV investment begins one period before the date of the *value1* cash flow and ends with the last cash flow in the list. The NPV calculation is based on future cash flows. If your first cash flow occurs at the beginning of the first period, the first value must be added to the NPV result, not included in the *values* arguments. For more information, see the examples below.
- If *n* is the number of cash flows in the list of *values*, the formula for NPV is:

$$\mathrm{NPV} = \sum_{i=1}^{n} \frac{values_i}{(1 + rate)^i}$$

- NPV is similar to the PV function (present value). The primary difference between PV and NPV is that PV allows cash flows to begin either at the end or at the beginning of the period. Unlike the variable NPV cash flow values, PV cash flows must be constant throughout the investment. For information on annuities and financial functions, see PV.
- NPV is also related to the IRR function (internal rate of return). IRR is the *rate* for which NPV equals zero: NPV(IRR(...),...)=0.

### Examples

Suppose you're considering an investment in which you pay $10,000 one year from today and receive an annual income of $3,000, $4,200, and $6,800 in the three years that follow. Assuming an annual discount rate of 10%, the net present value of this investment is:

NPV(10%,–10000,3000,4200,6800) equals $1,188.44

In the preceding example, you include the initial $10,000 cost as one of the *values*, because the payment occurs at the end of the first period.

Consider an investment that starts at the beginning of the first period. Suppose you're interested in buying a shoe store. The cost of the business is $40,000, and you expect to receive the following income for the first five years of operation: $8,000, $9,200, $10,000, $12,000, and $14,500. The annual discount rate is 8%; this might represent the rate of inflation, or the interest rate of a competing investment.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | Investment | ($40,000) | | |
| 2 | Year 1 Income | $8,000 | | |
| 3 | Year 2 Income | $9,200 | | |
| 4 | Year 3 Income | $10,000 | | |
| 5 | Year 4 Income | $12,000 | | |
| 6 | Year 5 Income | $14,500 | | |

The net present value of the shoe store investment is given by:

NPV(8%,B2:B6)+B1 equals $1,922.06

In the preceding example, you don't include the initial $40,000 cost as one of the *values*, because the payment occurs at the beginning of the first period.

Suppose your shoe store's roof collapses during the sixth year and you assume a loss of $9,000 for that year. The net present value of the shoe store investment after six years is given by:

NPV(8%,B2:B6,–9000)+B1 equals –$3,749.47

### Related Functions

| | |
|---|---|
| FV | Returns the future value of an investment |
| IRR | Returns the internal rate of return for an investment, without financing costs or reinvestment gains |
| PV | Returns the present value of an investment |

*Macro Sheets Only*

## OBJECT.PROTECTION(*locked,lock_text*)

Equivalent to the Format Object Protection command. Changes the protection status of the selected object. For detailed information on the Format Object Protection command, see online Help.

***Locked*** is a logical value that determines whether the selected object is locked or unlocked. If *locked* is TRUE, Microsoft Excel locks the object; if FALSE, Microsoft Excel unlocks the object.

***Lock_text*** is a logical value that determines whether text in a text box or button can be changed. *Lock_text* applies only if the object is a text box or button. If *lock_text* is TRUE or omitted, text cannot be changed; if FALSE, text can be changed.

### Notes

- You cannot lock or unlock an individual object with OBJECT.PROTECTION when document protection is turned on for objects in the Options Protect Document dialog box.
- If an object is not selected, the function returns the #VALUE! error value.
- In order for an object to be protected, you must use the PROTECT.DOCUMENT(,,TRUE) function after changing the object's status with OBJECT.PROTECTION.

### Related Function

| | |
|---|---|
| PROTECT.DOCUMENT | Turns on and off cell, window, and object protection |

## OFFSET(*reference,rows,cols,height,width*)

Returns a reference of a specified height and width, offset from another reference by a specified number of rows and columns. Use OFFSET in conjunction with the SELECT function or the *reference* argument to the FORMULA function to create a new selection of a specified size relative to the current selection.

### Note

OFFSET can now be used on worksheets; in earlier versions of Microsoft Excel it was a macro function only.

***Reference*** is the reference from which you want to base the offset. If *reference* is a multiple selection, OFFSET returns the #VALUE! error value.

***Rows*** is the number of rows, up or down, that you want the upper-left cell to refer to. Using 5 as the *rows* argument specifies that the upper-left cell in the reference is five rows below *reference*. *Rows* can be positive or negative.

***Cols*** is the number of columns, to the left or right, that you want the upper-left cell of the result to refer to. Using 5 as the *cols* argument specifies that the upper-left cell in the reference is five columns to the right of *reference*. *Cols* can be positive or negative.

If *rows* and *cols* offset *reference* over the edge of the worksheet, OFFSET returns the #REF! error value.

***Height*** is the height, in number of rows, that you want the returned reference to be. *Height* must be a positive number.

***Width*** is the width, in number of columns, that you want the returned reference to be. *Width* must be a positive number.

If *height* or *width* is omitted, it is assumed to be the same height or width as *reference*.

### Note

OFFSET doesn't actually move any cells or change the selection; it just returns a reference. To select a range offset from the current selection, use OFFSET with the SELECT and SELECTION functions. You can also select a cell offset from the current selection by using a relative reference with the SELECT function, for example, SELECT("R[1]C").

### Examples

OFFSET(C3,2,3,1,1) equals F5

OFFSET(C3:E5,–1,0,3,3) equals C2:E4

OFFSET(C3:E5,0,–3,3,3) equals #REF!

The following macro formula selects a range of the same size as the current selection one row down:

```
SELECT(OFFSET(SELECTION(),1,0))
```

The following macro formula puts the result of the DOCUMENTS function into a range with a height of one row and a width corresponding to the number of columns returned by the DOCUMENTS function, and offset from the active cell by one row and one column:

```
FORMULA.ARRAY(DOCUMENTS(),OFFSET(SELECTION(),1,
1,1,COLUMNS(DOCUMENTS())))
```

### Related Functions

| | |
|---|---|
| ABSREF | Returns the absolute reference of a cell |
| REFTEXT | Converts a reference into text |
| RELREF | Returns the relative reference of a cell |
| SELECT | Selects cells on a worksheet or an item on a chart |
| SELECTION | Returns the reference of the current selection |

*Macro Sheets Only*

## *ON* functions

ON functions allow you to specify a macro to be run when a certain event occurs. The ON functions turn on and off this special event handling.

ON functions are turned on by specifying the type of event to wait for, such as recalculation, a specific time, or a key to be pressed, and the macro to be run when the event occurs. ON functions are turned off by using the same formula but omitting the argument specifying the macro to be run.

For more information about specific ON functions, see their descriptions below.

*Macro Sheets Only*

## *ON.DATA(document_text,macro_text)*

Starts a specified macro when another application sends data to a particular document via DDE. DDE links to documents in other applications are called remote references. For more information about creating and using remote references, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*.

### Note

Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

***Document_text*** is the name of the document to which remote data will be sent or the name of the source of the remote data.

- If *document_text* is the name of the remote data, it must be in the form *app|topic!item*. *App*, *topic*, and *item* are each optional, but you must include the ! to indicate that the macro will be run when the source sends data.
- If *document_text* is omitted, the macro specified by *macro_text* is run whenever remote data is sent to Microsoft Excel.

***Macro_text*** is a name or R1C1-style reference to a macro that you want to run when data comes into the document or source specified by *document_text*. If *macro_text* is omitted, the ON.DATA function is turned off for the specified document or source.

### Notes

- ON.DATA remains in effect until you either turn it off or quit Microsoft Excel. You can turn off ON.DATA by specifying *document_text* and omitting the *macro_text* argument. If the macro sheet containing *macro_text* is closed when data is sent to *document_text*, an error is returned.
- If the incoming data causes recalculation, Microsoft Excel first runs the macro *macro_text* and then performs the recalculation.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula runs the macro AddOrders when data is sent to the worksheet ORDERSDB.XLS:

```
ON.DATA("ORDERSDB.XLS","AddOrders")
```

In Microsoft Excel for the Macintosh, the following macro formula runs the macro AddSale when data is sent to the worksheet Sales Database:

```
ON.DATA("Sales Database","AddSale")
```

The following macro formula runs the macro SentAll when Q+E sends all its Query1 data to Microsoft Excel:

```
ON.DATA("QE|Query1!ALL","SentAll")
```

### Related Functions

| | |
|---|---|
| ERROR | Specifies an action to take if an error occurs while a macro is running |
| INITIATE | Opens a channel to another application |
| ON.RECALC | Runs a macro when a document is recalculated |
| ON.TIME | Runs a macro at a specified time |
| ON.WINDOW | Runs a macro when a specified window is activated |

*Macro Sheets Only*

## *ON.KEY(key_text,macro_text)*

Runs a specified macro when a particular key or key combination is pressed.

***Key_text*** can specify any single key, or any key combined with ALT, CTRL, or SHIFT, or any combination of those keys (in Microsoft Excel for Windows or OS/2) or COMMAND, CTRL, OPTION, or SHIFT or any combination of those keys (in Microsoft Excel for the Macintosh). Each key is represented by one or more characters, such as "a" for the character a, or "{ENTER}" for the ENTER key.

To specify characters that aren't displayed when you press the key, such as ENTER or TAB, use the codes shown in the following table. Each code in the table represents one key on the keyboard.

| Key | Code |
|---|---|
| BACKSPACE | {BACKSPACE} or {BS} |
| BREAK | {BREAK} |
| CAPS LOCK | {CAPSLOCK} |
| CLEAR | {CLEAR} |
| DELETE or DEL | {DELETE} or {DEL} |
| DOWN | {DOWN} |
| END | {END} |
| ENTER | {ENTER} |
| ESC | {ESCAPE} or {ESC} |
| HELP | {HELP} |
| HOME | {HOME} |
| INS | {INSERT} |
| LEFT | {LEFT} |
| NUM LOCK | {NUMLOCK} |
| PAGE DOWN | {PGDN} |
| PAGE UP | {PGUP} |
| PRINT SCREEN | {PRTSC} |
| RETURN | {RETURN} or ~ (tilde) |
| RIGHT | {RIGHT} |
| SCROLL LOCK | {SCROLLLOCK} |
| TAB | {TAB} |
| UP | {UP} |
| F1 through F15 | {F1} through {F15} |

In Microsoft Excel for Windows or OS/2, you can also specify keys combined with SHIFT and/or CTRL and/or ALT. In Microsoft Excel for the Macintosh, you can also specify keys combined with SHIFT and/or CTRL and/or OPTION and/or COMMAND. To specify a key combined with another key or keys:

| To combine with | Precede the key code by |
|---|---|
| SHIFT | + (plus sign) |
| CTRL | ^ (caret) |
| ALT or OPTION | % (percent sign) |
| COMMAND | * (asterisk) |

To repeat a key sequence, use the form {*Key* number}. There must be a space between the key and the number. For example, {LEFT 42} means press the LEFT key 42 times.

***Macro_text*** is an R1C1-style reference to a macro that you want to run when *key_text* is pressed.

- If *macro_text* is "" (empty text), nothing happens when *key_text* is pressed. This form of ON.KEY disables the normal meaning of keystrokes in Microsoft Excel.
- If *macro_text* is omitted, *key_text* reverts to its normal meaning in Microsoft Excel. Use this form of ON.KEY to turn off any special key assignments made with previous ON.KEY functions.

### Note

ON.KEY remains in effect until you turn it off or quit Microsoft Excel. You can turn off ON.KEY by specifying *key_text* and omitting the *macro_text* argument. If the macro sheet containing *macro_text* is closed when you press *key_text*, an error is returned.

### Examples

Suppose you wanted the key combination SHIFT+CTRL+RIGHT to run the macro Print. You use the following macro formula:

```
ON.KEY("+^{RIGHT}","Print")
```

To return SHIFT+CTRL+RIGHT to its normal meaning, you would use the following macro formula:

```
ON.KEY("+^{RIGHT}")
```

To disable SHIFT+CTRL+RIGHT altogether, you would use the following macro formula:

```
ON.KEY("+^{RIGHT}","")
```

The following macro runs the UNLOCKED.NEXT function whenever ENTER is pressed. The ON.KEY function assigns a macro to run when ENTER is pressed. The ResetEnterKey macro returns ENTER to its normal function. Use this example when you want ENTER to automatically move to the next cell on a protected sheet.



| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | SetEnterKey | |
| 4 | | =ON.KEY("~","EnterNext") | Runs EnterNext on ENTER |
| 5 | | =ON.KEY("^~","EnterPrev") | Runs EnterPrev on CTRL+ENTER |
| 6 | | =RETURN() | |
| 7 | | | |
| 8 | | | |
| 9 | EnterNext | =UNLOCKED.NEXT() | Goes to next unlocked cell |
| 10 | | =RETURN() | |
| 11 | | | |
| 12 | | | |
| 13 | EnterPrev | =UNLOCKED.PREV() | Goes to previous unlocked cell |
| 14 | | =RETURN() | |
| 15 | | | |
| 16 | | | |
| 17 | ResetEnter | =ON.KEY("~") | Returns ENTER to normal |
| 18 | | =ON.KEY("%~") | Returns CTRL+ENTER to normal |
| 19 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| CANCEL.KEY | Specifies an action to take when a cancel key is pressed |
| ERROR | Specifies an action to take if an error occurs while a macro is running |
| ON.DATA | Runs a macro when data is sent to Microsoft Excel by another application |
| ON.RECALC | Runs a macro when a document is recalculated |
| ON.TIME | Runs a macro at a specified time |
| ON.WINDOW | Runs a macro when a specified window is activated |

*Macro Sheets Only*

## ON.RECALC(*sheet_text,macro_text*)

Runs a macro when a specific document is recalculated. Use ON.RECALC to perform an operation, such as checking that a certain condition is still being met, on a document each time the document is recalculated.

***Sheet_text*** is the name of a document, given as quoted text. If *sheet_text* is omitted, the macro is run whenever any open document not specified by a previous ON.RECALC formula is recalculated. Only one ON.RECALC formula can be run for each recalculation.

***Macro_text*** is the name or R1C1-style reference of a macro you want to run when the worksheet specified by *sheet_text* is recalculated. The macro will be run each time the document is recalculated until ON.RECALC is turned off.

### Note

A macro specified to be run by ON.RECALC is not run by actions taken by other macros. For example, a macro specified by ON.RECALC will not be run after the CALCULATE.NOW function is carried out, but will be run if you change data in a sheet set to calculate automatically, or choose the Options Calculate Now command.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula specifies that the macro Printer on the macro sheet AUTOREPT.XLM be run when the document named REPORT.XLS is recalculated:

```
ON.RECALC("REPORT.XLS","AUTOREPT.XLM!Printer")
```

In Microsoft Excel for the Macintosh, the following macro formula turns off ON.RECALC for the document named Sales:

```
ON.RECALC("Sales")
```

The following macro checks a particular value every time the worksheet is recalculated, and alerts the user if the value exceeds a certain amount:

| | B | C |
|---|---|---|
| 1 | *Formulas* | *Comments* |
| 2 | | |
| 3 | RecalcCheck | |
| 4 | =ON.RECALC("BUDGET.XLS","CheckTotal") | Checks the total |
| 5 | =RETURN() | |
| 6 | | |
| 7 | | |
| 8 | CheckTotal | |
| 9 | =IF(BUDGET.XLS!TotalMisc<10000,,ALERT("You are ove | Alert user if value is too high |
| 10 | =RETURN() | |

The complete formula in cell B8 is:

```
=IF(BUDGET.XLS!TotalMisc<10000,,ALERT("You are over budget for miscellaneous expenses.",3))
```

### Related Functions

| | |
|---|---|
| CALCULATE.DOCUMENT | Calculates the active document only |
| CALCULATE.NOW | Calculates all open documents immediately |
| CALCULATION | Controls calculation settings |

Macro Sheets Only

## ON.TIME(time,macro_text,tolerance,insert_logical)

Runs a macro at a specified time. Use ON.TIME to run a macro at a specific time of day or after a specified period has passed.

***Time*** is the time and date, given as a serial number, at which the macro is to be run. If *time* only specifies a time (that is, if *time* is a serial number less than 1), the macro *macro_text* runs every day at that time.

***Macro_text*** is the name or R1C1-style reference of a macro to run at the specified time.

***Tolerance*** is the time and date, given as a serial number, that you are willing to wait until and still have the macro run. For example, if Microsoft Excel is not in Ready, Copy, Cut, or Find mode at *time*, because another macro is running, but is in Ready mode 15 seconds later, and *tolerance* is set to *time* plus 30 seconds, the macro specified by *macro_text* will run. If Microsoft Excel was not in Ready mode within 30 seconds, the macro would not run. If *tolerance* is omitted, it is assumed to be infinite.

***Insert_logical*** is a logical value specifying whether you want *macro_text* to run at *time*. Use *insert_logical* when you want to turn off a previously set ON.TIME formula. If *insert_logical* is TRUE or omitted, the macro specified by *macro_text* is carried out at *time*. If *insert_logical* is FALSE and *macro_text* is not set to run at *time*, ON.TIME returns the #VALUE! error value.

### Examples

The following macro formula runs a macro called Test every day at 5:00:00 P.M. when Microsoft Excel is in Ready mode:

```
ON.TIME("5:00:00 PM","Test")
```

The following macro formula runs a macro called Test 5 seconds after the formula is evaluated:

```
ON.TIME(NOW()+"00:00:05","Test")
```

The following macro formula runs a macro called Test 10 seconds after the formula is evaluated. If Microsoft Excel is not in Ready mode at that time (because it is in Edit mode, for example), the *tolerance* argument specifies 5 seconds of additional time to wait to run the macro. If Microsoft Excel is still not in Ready mode at that time, *macro_text* is not run.

```
ON.TIME(NOW()+"00:00:10","Test",NOW()+"00:00:15")
```

### Related Function

| | |
|---|---|
| NOW | Returns the serial number of the current date and time |

Macro Sheets Only

## ON.WINDOW(window_text,macro_text)

Starts a specified macro when a particular window is activated.

***Window_text*** is the name of a window in the form of text. If *window_text* is omitted, ON.WINDOW starts the macro whenever any window is activated, except for windows that are named in other ON.WINDOW statements.

***Macro_text*** is the name or R1C1-style reference to a macro to run when *window_text* is activated. If *macro_text* is omitted, activating *window_text* no longer starts a macro.



### Note

A macro specified to be run by ON.WINDOW is not run by actions taken by other macros. For example, a macro specified by ON.WINDOW will not be run after the ACTIVATE(*window_text*) function is carried out, but will be run if you activate the window by clicking it or by using the list on the Window menu.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula runs the macro named ShowAlert when the window MAIN.XLM is activated:

```
ON.WINDOW("MAIN.XLM","ShowAlert")
```

The following macro formula stops the macro from running when MAIN.XLM is activated:

```
ON.WINDOW("MAIN.XLM")
```

In Microsoft Excel for the Macintosh, the following macro formula runs the macro named ShowAlert when the window Main Window is activated:

```
ON.WINDOW("Main Window","ShowAlert")
```

The following macro formula stops the macro from running when Main Window is activated:

```
ON.WINDOW("Main Window")
```

### Related Functions

| | |
|---|---|
| GET.WINDOW | Returns information about a window |
| ON.DATA | Runs a macro when data is sent to Microsoft Excel by another application |
| ON.KEY | Runs a macro when a specified key is pressed |
| ON.TIME | Runs a macro at a specified time |
| WINDOWS | Returns a list of the open windows |

Macro Sheets Only

## OPEN(file_text,update_links,read_only,format,prot_pwd,write_res_pwd,ignore_rorec,file_origin)

## OPEN?(file_text,update_links,read_only,format,prot_pwd,write_res_pwd,ignore_rorec,file_origin)

Equivalent to the File Open command. Opens an existing file or workspace. For detailed information on the File Open command, see online Help.

***File_text*** is the name, as text, of the document you want to open. *File_text* can include a drive and pathname. In the dialog-box form, OPEN?, *file_text* can include an asterisk (*) to represent any sequence of characters and a question mark (?) to represent any single character.

***Update_links*** specifies whether and how to update external and remote references. If *update_links* is omitted, Microsoft Excel displays a message asking if you want to update links.

| If *update_links* is | Then |
|---|---|
| 0 | Update neither external nor remote references |
| 1 | Update external references only |
| 2 | Update remote references only |
| 3 | Update external and remote references |

### Note

When you are opening a file in WKS or WK1 format, the *update_links* argument specifies whether Microsoft Excel generates charts from any graphs attached to the WKS or WK1 file.

| If *update_links* is | Charts are |
|---|---|
| 0 | Not created |
| 2 | Created |

***Read_only*** corresponds to the Read Only check box in the File Open dialog box. If *read_only* is TRUE, the document can be modified but changes cannot be saved; if FALSE or omitted, changes to the document can be saved.

***Format*** specifies whether the values in a text file are separated by tabs or commas. If *format* is omitted, Microsoft Excel uses the value currently set in the File Open dialog box.

| If *format* is | Values are separated by |
|---|---|
| 1 | Tabs |
| 2 | Commas |

***Prot_pwd*** is the password, as text, required to unprotect a protected file. If *prot_pwd* is omitted and *file_text* requires a password, the Password dialog box is displayed. Passwords are case sensitive. Passwords are not recorded when you open a document and supply the password with the macro recorder on.

***Write_res_pwd*** is the password, as text, required to open a read-only file with full write privileges. If *write_res_pwd* is omitted and *file_text* requires a password, the Password dialog box is displayed.

***Ignore_rorec*** is a logical value that controls whether the read-only recommended message is displayed. If *ignore_rorec* is TRUE, Microsoft Excel prevents display of the message; if FALSE or omitted, and if *read_only* is FALSE or omitted, Microsoft Excel displays the alert when opening a read-only recommended document.

***File_origin*** is a number specifying whether a text file originated on the Macintosh, Windows, or OS/2.

| File_origin | Original operating environment |
|---|---|
| 1 | Macintosh |
| 2 | Windows (ANSI) |
| 3 | DOS or OS/2 (PC-8) |
| Omitted | Current operating environment |

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula opens the file SALES.XLS as read only, updates both external and remote references, and supplies the password "Frankly":

```
OPEN("SALES.XLS",3,TRUE,,"Frankly")
```

The following formula opens the file SALES.XLS in the specified path:

```
OPEN("C:\SYSTEM\EXCEL\SALES.XLS")
```

In Microsoft Excel for the Macintosh, the following formula opens the file List Data with edit privileges, updates no references, specifies the data as comma separated, and supplies the password "Manage1":

```
OPEN("List Data",0,FALSE,2,"Manage1")
```

The following formula opens the file List Data in the specified path:

```
OPEN("Hard Disk:Excel:List Data")
```

### Related Function

| | |
|---|---|
| FOPEN | Opens a text file |
| OPEN.LINKS | Opens specified supporting documents |

*Macro Sheets Only*

## OPEN.LINKS(*document_text1,document_text2,..., read_only,type_of_link*)

## OPEN.LINKS?(*document_text1, document_text2,...,read_only,type_of_link*)

Equivalent to the File Links command (full menus). Use OPEN.LINKS with the LINKS function to open documents linked to a particular document. For detailed information on the File Links command, see online Help.

***Document_text1,document_text2...*** are 1 to 12 arguments that are the names of supporting documents in the form of text, or arrays or references that contain text.

***Read_only*** corresponds to the Read Only check box in the File Links dialog box. If *read_only* is TRUE, the document can be modified but changes cannot be saved; if FALSE or omitted, changes to the document can be saved. *Read_only* applies only to Microsoft Excel, WKS, and SYLK documents.

***Type_of_link*** is a number from 1 to 6 that specifies what type of link you want to get information about.



| Type_of_link | Link document type |
|---|---|
| 1 | Microsoft Excel link |
| 2 | DDE link |
| 3 | Reserved |
| 4 | Outgoing NowWave link |
| 5 | Subscriber |
| 6 | Publisher |

### Note

You can generate an array of the names of linked documents with the LINKS function.

### Examples

In Microsoft Excel for Windows or OS/2, if a chart named PERFORM.XLC is open and contains data series that refer to worksheets named BUDGET.XLS and ACTUAL.XLS, these worksheets are opened when this formula is calculated:

```
OPEN.LINKS(LINKS())
```

In Microsoft Excel for the Macintosh, if a chart named Performance Chart is open and contains data series that refer to worksheets named Budget and Actual, these worksheets are opened when this formula is calculated:

```
OPEN.LINKS(LINKS())
```

### Related Functions

| | |
|---|---|
| CHANGE.LINK | Changes supporting worksheet links |
| GET.LINK.INFO | Returns information about a link |
| LINKS | Returns the name of all linked documents |
| UPDATE.LINK | Updates a link |

*Macro Sheets Only*

## OPEN.MAIL(*subject,comments*)

## OPEN.MAIL?(*subject,comments*)

Equivalent to the File Open Mail command. Opens files sent via Microsoft Mail that Microsoft Excel can open. For detailed information on the File Open Mail command, see online Help.

### Note

This function is only available if you are using Microsoft Excel for the Macintosh and Microsoft Mail version 2.0 or later.

***Subject*** is the subject of the message containing a file that Microsoft Excel can open. For each message whose subject matches the *subject* argument and contains a file that Microsoft Excel can open, the file is opened in Microsoft Excel; if the message has no unread enclosures, it is deleted from the list of pending mail. If *subject* is omitted, then for all messages containing files Microsoft Excel can open, the files are opened and each message that has no unread enclosures is deleted.

***Comments*** is a logical value that specifies whether comments associated with the Microsoft Excel files are displayed. If *comments* is TRUE, Microsoft Excel displays the comments; if FALSE, comments are not displayed.

### Tips

- If you use consistent subjects in your Microsoft Mail messages, you can easily create a macro that always opens mail messages with certain files attached. For example, an OPEN.MAIL formula with *subject* specified as "Weekly Report" would open the Microsoft Excel file attached to the message containing that subject each week.
- In OPEN.MAIL?, the dialog-box form of the function, the currently running macro pauses while the Microsoft Mail documents window is displayed. Macro running resumes after you close the Microsoft Mail documents window.

### Examples

The following macro formula opens the Microsoft Excel file attached to the Microsoft Mail message with the subject heading "Payroll Data" without displaying comments:

OPEN.MAIL("Payroll Data",FALSE)

The following macro formula displays the Microsoft Mail documents window and all messages readable by Microsoft Excel:

OPEN.MAIL?()

### Related Function

| | |
|---|---|
| SEND.MAIL | Sends the active document or workgroup |

## OR(*logical1,logical2,...*)

Returns TRUE if one or more of the arguments is TRUE; returns FALSE if all arguments are FALSE.

***Logical1, logical2,...*** are 1 to 14 conditions you want to test that can be either TRUE or FALSE.

- The arguments should be logical values, or arrays or references that contain logical values.
- If an array or reference argument contains text or empty cells, those values are ignored.
- If the specified range contains no logical values, OR returns the #VALUE! error value.

### Examples

OR(TRUE) equals TRUE

OR(1+1=1,2+2=5) equals FALSE

If A1:A3 contains the values TRUE, FALSE, and TRUE, then:

OR(A1:A3) equals TRUE

The following macro formula checks the contents of the active cell. If the cell contains the single character "c" or "s", the OR formula returns TRUE and the macro branches to an area named FinishRefresh:

IF(OR(ACTIVE.CELL()="c",ACTIVE.CELL()="s"),
GOTO(FinishRefresh))

The preceding example shows how to use the worksheet form of the IF function on a macro sheet. You could also use the macro sheet form. For more information, see IF.

Also see the example in EXACT.

### Tip

You can use an OR array formula to see if a value occurs in an array. For example, the following macro uses Microsoft Excel's type conversion feature to compare a given account number to a list of account numbers. The OR formula in cell B5 is entered as an array formula; this causes Microsoft Excel to expand the account number into an array of the same size as the list of account numbers. This is useful if you want to see if a value occurs anywhere in an array. If the account number is in the list, the macro calls a subroutine to check the account activity; otherwise, an error message is displayed.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **CheckAccount** | |
| 4 | AccountNum | =INPUT("Enter the account number",2) | Asks for the account number |
| 5 | InTheList | =OR(AccountNum=List) | Checks the account list |
| 6 | | =IF(InTheList) | If it's in the list, then... |
| 7 | | =CheckAccountActivity(AccountNum) | Calls subroutine |
| 8 | | =ELSE() | |
| 9 | | =ALERT("The account number is not in the li | Otherwise displays message |
| 10 | | =END.IF() | and ends |
| 11 | | =RETURN() | |

The complete formula in cell B9 is:

```
=ALERT("The account number is not in the list",2)
```

### Related Functions

| | |
|---|---|
| AND | Returns TRUE if all its arguments are TRUE |
| NOT | Reverses the logic of its argument |



*Macro Sheets Only*

## OUTLINE(*auto_styles,row_dir,col_dir,create_apply*)

Corresponds to the Formula Outline command. Creates an outline and defines settings for automatically creating outlines. For detailed information on the Formula Outline command, see online Help.

The first three arguments are logical values corresponding to check boxes in the Formula Outline dialog box. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box. If an argument is omitted, the check box is left in its current state.

***Auto_styles*** corresponds to the Automatic Styles check box.

***Row_dir*** corresponds to the Summary Rows Below Detail check box.

***Col_dir*** corresponds to the Summary Columns To Right Of Detail check box.

***Create_apply*** is the number 1 or 2 and corresponds to the Create button and the Apply Styles button.

| *Create_apply* | Result |
|---|---|
| 1 | Creates an outline with the current settings |
| 2 | Applies outlining styles to the selection based on outline levels |

### Example

The following macro formula creates an outline, turns on automatic outlining styles, and assumes that the summary columns and rows are to the right and below the detail, respectively:

OUTLINE(TRUE,TRUE,TRUE,1)

### Related Functions

| | |
|---|---|
| DEMOTE | Demotes selected rows or columns in an outline |
| PROMOTE | Promotes selected rows or columns in an outline |

*Macro Sheets Only*

## OVERLAY(*type_num,stack,100,vary,overlap,drop,hilo,overlap%,cluster,angle,series_num,auto*)

## OVERLAY?(*type_num,stack,100,vary,overlap,drop,hilo,overlap%,cluster,angle,series_num,auto*)

Equivalent to the Format Overlay command in Microsoft Excel version 2.2 or earlier. This function is included only for macro compatibility. To format an overlay chart in Microsoft Excel version 3.0, use the FORMAT.OVERLAY function, which is equivalent to the Format Overlay command.

*Macro Sheets Only*

## OVERLAY.CHART.TYPE(*type_num*)

Equivalent to the Chart Overlay Chart Type command in Microsoft Excel for the Macintosh version 1.5 or earlier. This function is included only for macro compatibility. To format an overlay chart or to specify an overlay chart type in Microsoft Excel version 3.0, use the FORMAT.OVERLAY function, which is equivalent to the Format Overlay command.

*Macro Sheets Only*

## PAGE.SETUP(*head,foot,left,right,top,bot,heading,grid,h_center,v_center,orientation,paper_size,scaling*)

## PAGE.SETUP?(*head,foot,left,right,top,bot,heading,grid,h_center,v_center,orientation,paper_size,scaling*)

**Form 1: Worksheets and macro sheets**

## PAGE.SETUP(*head,foot,left,right,top,bot,size,h_center,v_center,orientation,paper_size,scaling*)

## PAGE.SETUP?(*head,foot,left,right,top,bot,size,h_center,v_center,orientation,paper_size,scaling*)

**Form 2: Charts**

Equivalent to the File Page Setup command. Use PAGE.SETUP to control the printed appearance of

your documents. For detailed information on the File Page Setup command, see online Help.

There are two forms of PAGE.SETUP. The first form applies if the active document is a worksheet or macro sheet; the second form applies if the active document is a chart.

Arguments correspond to check boxes and text boxes in the File Page Setup dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Head*** corresponds to the Header box and is text specifying the header for the document.

***Foot*** corresponds to the Footer box and is text specifying the footer for the document.

***Left*** corresponds to the Left box and is a number specifying the left margin.

***Right*** corresponds to the Right box and is a number specifying the right margin.

***Top*** corresponds to the Top box and is a number specifying the top margin.

***Bot*** corresponds to the Bottom box and is a number specifying the bottom margin.

***Heading*** corresponds to the Row & Column Headings check box. *Heading* is available only in the worksheet and macro sheet form of the function.

***Grid*** corresponds to the Gridlines check box. *Grid* is available only in the worksheet and macro sheet form of the function.

***Size*** is a number corresponding to the options in the Size box, and determines how you want the chart printed on the page within the margins. *Size* is available only in the chart form of the function.

| *Size* | Size to print the chart |
|---|---|
| 1 | Screen size |
| 2 | Fit to page |
| 3 | Full page |

***H_center*** corresponds to the Center Horizontally check box in the File Page Setup dialog box.

***V_center*** corresponds to the Center Vertically check box in the File Page Setup dialog box.

The next three arguments are only available in Microsoft Excel for Windows and depend on your printer and printer driver. They are ignored in other versions of Microsoft Excel.

***Orientation*** determines the direction in which your document is printed.

| *Orientation* | Print format |
|---|---|
| 1 | Portrait |
| 2 | Landscape |

***Paper_size*** is a number from 1 to 26 that specifies the size of the paper.

| *Paper_size* | Paper type |
|---|---|
| 1 | Letter |
| 2 | Letter (small) |
| 3 | Tabloid |
| 4 | Ledger |
| 5 | Legal |
| 6 | Statement |
| 7 | Executive |
| 8 | A3 |
| 9 | A4 |
| 10 | A4 (small) |
| 11 | A5 |
| 12 | B4 |
| 13 | B5 |
| 14 | Folio |
| 15 | Quarto |
| 16 | 10x14 |
| 17 | 11x17 |
| 18 | Note |



| *Paper_size* | Paper type |
|---|---|
| 19 | ENV9 |
| 20 | ENV10 |
| 21 | ENV11 |
| 22 | ENV12 |
| 23 | ENV14 |
| 24 | C Sheet |
| 25 | D Sheet |
| 26 | E Sheet |

***Scaling*** is a number representing the percentage to increase or decrease the size of the document. *Scaling* is ignored when using printers that do not provide scaling. You can fit the print area on a single page by using TRUE as the *scaling* argument. This corresponds to the Fit To Page check box in the File Page Setup dialog box.

### Related Functions

| | |
|---|---|
| DISPLAY | Controls how an object is displayed |
| PRINT | Prints the active document |
| WORKSPACE | Changes the workspace settings for a document |

*Macro Sheets Only*

## PARSE(*text*)
## PARSE?(*text*)

Equivalent to the Data Parse command (full menus). Distributes the contents of a single column to fill several adjacent columns. Use PARSE to reorganize data, especially data that you've read from files created by another application, such as a database.

***Text*** is the parse line in the form of text. It is a copy of the first nonblank cell in the selected column, with square brackets indicating where to distribute (or parse) text.

### Notes

- In most cases, it's easier to record the Data Parse command than to type the PARSE function manually.
- Before you use the PARSE function, you must have already selected a range no more than one column wide.
- When you use the PARSE function or choose the Data Parse command, Microsoft Excel splits the first column into as many columns as you specify with *text* and overwrites any information in the adjacent columns.

### Example

Suppose you have opened a text file and all the information has been read into a single column. If the delimiter (separator) character occurs at regularly spaced intervals, you can select the column and use PARSE. If the delimiter character occurs randomly in each cell, you can use the following macro to parse each cell into several cells.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **ParseDelimiterAnywhere** | Parses a range of cells |
| 4 | | =ECHO(FALSE) | Screen updating off |
| 5 | | =IF(COLUMNS(SELECTION())>1,AlertArea()) | More than one column? |
| 6 | Separator | =INPUT("Enter the separator character",3) | Gets parse separator |
| 7 | | FOR CELL("ThatCell",SELECTION()) | Begins loop over the range |
| 8 | | SELECT(ThatCell) | Selects the next cell to parse |
| 9 | CellCont | =GET.CELL(6,ThatCell) | Gets contents of active cell |
| 10 | | IF(ISERROR(SEARCH(Separator,CellCont,1))) | Looks for separator |
| 11 | | [=CellCont&"] | Puts brackets around text |
| 12 | ParseText | =CELL("width",ThatCell)-LEN(CellCont)+1 | Replaces Separator with brackets |
| 13 | | =PARSE(ParseText) | Parses the data |
| 14 | | =END.IF() | |
| 15 | | =NEXT() | |
| 16 | | =RETURN() | |
| 17 | | | |
| 18 | | | |
| 19 | | **AlertArea** | |
| 20 | | =ALERT("You must have only one column.",1) | |
| 21 | | =HALT() | |

The complete formula in cell B10 is:

```
=IF(NOT(ISERROR(SEARCH(Separator,CellCont,1))))
```

If you want to specify a space as the delimiter, replace the variable Separator with a space enclosed in quotation marks and remove row 5.

***Related Functions***

| | |
|---|---|
| MID | Extracts characters based on a starting position and number of characters |
| SEARCH | Finds one text value within another (not case sensitive) |

*Macro Sheets Only*

## PASTE( )

Equivalent to the Edit Paste command. Pastes a selection or object that you copied or cut using the COPY or CUT function. Use PASTE when you want to paste all components of the copy area. To paste only specific components of the copy area, use PASTE.SPECIAL. For detailed information on the Edit Paste command, see online Help.

***Tips***

- You can use PASTE to create charts composed of stacked pictures, such as stacked coins to show different amounts of money, or stretched pictures, such as taller or shorter people to show changes in population. The picture can be created using Microsoft Excel's drawing tools or created in and copied from a separate drawing or painting application. Microsoft Excel associates the picture with the selected data series or with the selected data point if the chart has only one series.
- For information on changing stacked and stretched picture formats, see form 8 of the PATTERNS function later in this book. For information on creating charts with stacked and stretched pictures, see Chapter 13, "Formatting a Chart," in the *Microsoft Excel User's Guide*.

***Example***

See the example for SHOW.CLIPBOARD.

***Related Functions***

| | |
|---|---|
| COPY | Copies information to the Clipboard |
| CUT | Defines the selection that will be moved when you use PASTE |
| FORMULA | Enters a formula in a cell or text on a chart |
| PASTE.LINK | Pastes copied data and establishes a link to its source |
| PASTE.SPECIAL | Pastes specific components of copied data |

*Macro Sheets Only*

## PASTE.LINK( )

Equivalent to the Edit Paste Link command. Pastes copied data or objects and establishes a link with the source of the data or object. The source can be either another Microsoft Excel worksheet or another application. Use PASTE.LINK when you want Microsoft Excel to automatically update the paste area with any changes that occur in the source. For detailed information on the Edit Paste Link command, see online Help.

***Note***

To work properly, the application you are linking to must support Dynamic Data Exchange (DDE). For more information, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*.



***Related Functions***

| | |
|---|---|
| COPY | Copies information to the Clipboard |
| CUT | Defines the selection that will be moved when you use PASTE |
| PASTE | Pastes copied or cut data |
| PASTE.SPECIAL | Pastes specific components of copied data |

*Macro Sheets Only*

## PASTE.PICTURE( )

Equivalent to the Edit Paste Picture command. Pastes a picture of the Clipboard contents onto the worksheet at the active cell. This picture is not linked, so changes to the source data will not be reflected in the picture. For detailed information on the Edit Paste Picture command, see online Help.

***Related Functions***

| | |
|---|---|
| COPY.PICTURE | Creates a picture of the selection |
| PASTE.PICTURE.LINK | Pastes a linked picture of the currently copied area |

*Macro Sheets Only*

## PASTE.PICTURE.LINK( )

Equivalent to the Edit Paste Picture Link command and to using the camera tool on the tool bar. Pastes a linked picture of the Clipboard contents onto the worksheet at the active cell. This picture is linked, so changes to the source data will be reflected in the picture. For detailed information on the Edit Paste Picture Link command, see online Help.

***Related Functions***

| | |
|---|---|
| COPY.PICTURE | Creates a picture of the selection |
| CREATE.OBJECT | Creates the specified object |
| PASTE.PICTURE | Pastes a picture of the currently copied area |

*Macro Sheets Only*

## PASTE.SPECIAL(*paste_num,operation_num, skip_blanks,transpose*)

## PASTE.SPECIAL?(*paste_num,operation_num, skip_blanks,transpose*)

**Form 1: From a worksheet to a worksheet**

Equivalent to the Edit Paste Special command (full menus). Pastes the specified components from the copy area into the current selection. The PASTE.SPECIAL function has three forms. Use this form if you are pasting into a worksheet or macro sheet.

***Paste_num*** is a number from 1 to 5 specifying what to paste.

| *Paste_num* | Pastes |
|---|---|
| 1 | All |
| 2 | Formulas |
| 3 | Values |
| 4 | Formats |
| 5 | Notes |

***Operation_num*** is a number from 1 to 5 specifying which operation to perform when pasting.

| *Operation_num* | Action |
|---|---|
| 1 | None |
| 2 | Add |
| 3 | Subtract |
| 4 | Multiply |
| 5 | Divide |

***Skip_blanks*** is a logical value corresponding to the Skip Blanks check box in the Edit Paste Special dialog box.

- If *skip_blanks* is TRUE, Microsoft Excel skips blanks in the copy area when pasting.
- If *skip_blanks* is FALSE, Microsoft Excel pastes normally.

***Transpose*** is a logical value corresponding to the Transpose check box in the Edit Paste Special dialog box.

- If *transpose* is TRUE, Microsoft Excel transposes rows and columns when pasting.
- If *transpose* is FALSE, Microsoft Excel pastes normally.

### Example

The following macro formula pastes all formulas, values, formats, and notes, and it subtracts the numeric contents of the Clipboard from the numbers in the current selection. It also skips blank cells.

```
PASTE.SPECIAL(1,3,TRUE,FALSE)
```

### Related Functions

| | |
|---|---|
| FORMULA | Enters a formula in a cell or text on a chart |
| PASTE | Pastes copied data |
| PASTE.LINK | Pastes copied data and establishes a link to its source |

Macro Sheets Only

## PASTE.SPECIAL(*rowcol,series,categories,replace*)
## PASTE.SPECIAL?(*rowcol,series,categories,replace*)

**Form 2: From a worksheet to a chart**

Equivalent to the Edit Paste Special command on the chart menu bar (full menus). Pastes the specified components from the copy area into a chart. The PASTE.SPECIAL function has three forms. Use this form if you have copied from a worksheet and are pasting into a chart.

***Rowcol*** is the number 1 or 2 and specifies whether the values corresponding to a particular data series are in rows or columns.

| *Rowcol* | Pastes values in |
|---|---|
| 1 | Rows |
| 2 | Columns |

***Series*** is a logical value corresponding to the Series Names In First Column (or First Row, depending on the value of *rowcol*) check box in the Edit Paste Special dialog box.

- If *series* is TRUE, Microsoft Excel turns on the check box and uses the contents of the cell in the first column of each row (or first row of each column) as the name of the data series in that row (or column).
- If *series* is FALSE, Microsoft Excel turns off the check box and uses the contents of the cell in the first column of each row (or first row of each column) as the first data point of the data series.

***Categories*** is a logical value corresponding to the Categories (X Labels) In First Row (or First Column, depending on the value of *rowcol*) check box in the Edit Paste Special dialog box.

- If *categories* is TRUE, Microsoft Excel turns on the check box and uses the contents of the first row (or column) of the selection as the categories for the chart.
- If *categories* is FALSE, Microsoft Excel turns off the check box and uses the contents of the first row (or column) as the first data series in the chart.

***Replace*** is a logical value corresponding to the Replace Existing Categories check box in the Edit Paste Special dialog box.

- If *replace* is TRUE, Microsoft Excel turns on the check box and applies categories while replacing existing categories with information from the copied cell range.
- If *replace* is FALSE, Microsoft Excel turns off the check box and applies new categories without replacing any old ones.



### Example

The following macro formula pastes to a chart. Values are in rows, with series names in the first column:

```
PASTE.SPECIAL(1,TRUE,FALSE,FALSE)
```

Macro Sheets Only

## PASTE.SPECIAL(*paste_num*)
## PASTE.SPECIAL?(*paste_num*)

**Form 3: From a chart to a chart**

Equivalent to the Edit Paste Special command (full menus). Pastes the specified components from the copy area into a chart. The PASTE.SPECIAL function has three forms. Use this form if you have copied from a chart and are pasting into a chart.

***Paste_num*** is a number from 1 to 3 specifying what to paste.

| *Paste_num* | Pastes |
|---|---|
| 1 | All (formats and data series) |
| 2 | Formats only |
| 3 | Formulas (data series) only |

Macro Sheets Only

## PATTERNS(*apattern,afore,aback*)
## PATTERNS?(*apattern,afore,aback*)

**Form 1: Cells**

## PATTERNS(*lauto,lstyle,lcolor,lwt,hwidth,hlength,htype*)
## PATTERNS?(*lauto,lstyle,lcolor,lwt,hwidth,hlength,htype*)

**Form 2: Lines (arrows) on worksheets or charts**

## PATTERNS(*bauto,bstyle,bcolor,bwt,shadow,aauto,apattern,afore,aback,rounded*)
## PATTERNS?(*bauto,bstyle,bcolor,bwt,shadow,aauto,apattern,afore,aback,rounded*)

**Form 3: Text boxes, rectangles, ovals, arcs, and pictures**

## PATTERNS(*bauto,bstyle,bcolor,bwt,shadow,aauto,apattern,afore,aback,invert,apply*)
## PATTERNS?(*bauto,bstyle,bcolor,bwt,shadow,aauto,apattern,afore,aback,invert,apply*)

**Form 4: Chart plot areas, bars, columns, pie slices, and text labels**

## PATTERNS(*lauto,lstyle,lcolor,lwt,tmajor,tminor,tlabel*)
## PATTERNS?(*lauto,lstyle,lcolor,lwt,tmajor,tminor,tlabel*)

**Form 5: Chart axes**

## PATTERNS(*lauto,lstyle,lcolor,lwt,apply*)
## PATTERNS?(*lauto,lstyle,lcolor,lwt,apply*)

**Form 6: Chart gridlines, hi-lo lines, drop lines, lines on a picture line chart, and picture charts of bar, column, and 3-D column charts**

## PATTERNS(*lauto,lstyle,lcolor,lwt,mauto,mstyle,mfore,mback,apply*)
## PATTERNS?(*lauto,lstyle,lcolor,lwt,mauto,mstyle,mfore,mback,apply*)

**Form 7: Chart data lines**

## PATTERNS(*type,picture_units,apply*)
## PATTERNS?(*type,picture_units,apply*)

**Form 8: Picture chart markers**

Equivalent to the Format Patterns command. Changes the appearance of the selected cells or objects or the selected chart item (you can select only one chart item at a time). The PATTERNS function has ten forms: Form 1 is for cells on a worksheet or macro sheet. Forms 2 and 3 are for objects on a worksheet or macro sheet. Forms 4 through 8 are for chart items. The exception is form 2, which can be used for any line or arrow, including those on a chart.

The following argument descriptions are in alphabetical order. Arguments correspond to check boxes, list boxes, and options in the Format Patterns dialog box for the selected item. The default for each argument reflects the setting in the dialog box.

***Aauto*** is the number 1 or 2 specifying area settings (that is, the object's "surface area").

| If *aauto* is | Area settings are |
|---|---|
| 0 | Set by the user (custom) |
| 1 | Automatic (set by Microsoft Excel) |
| 2 | None |

***Aback*** is a number from 1 to 16 corresponding to the 16 area background colors in the Format Patterns dialog box.

***Afore*** is a number from 1 to 16 corresponding to the 16 area foreground colors in the Format Patterns dialog box.

***Apattern*** is a number from 1 to 18 corresponding to the 18 area patterns in the Format Patterns dialog box. If *apattern* is 0 and a cell is selected, Microsoft Excel applies no pattern.

***Apply*** is a logical value corresponding to the Apply To All check box. This argument applies only when a chart data point or a data series is selected.

- If *apply* is TRUE, Microsoft Excel applies any formatting changes to all items that are similar to the selected item on the chart.
- If *apply* is FALSE, Microsoft Excel applies formatting changes only to the selected item on the chart.

***Bauto*** is a number from 0 to 2 specifying border settings.

| If *bauto* is | Border settings are |
|---|---|
| 0 | Set by the user (custom) |
| 1 | Automatic (set by Microsoft Excel) |
| 2 | None |

***Bcolor*** is a number from 1 to 16 corresponding to the 16 border colors in the Format Patterns dialog box.

***Bstyle*** is a number from 1 to 8 corresponding to the eight border styles in the Format Patterns dialog box.

***Bwt*** is a number from 1 to 4 corresponding to the four border weights in the Format Patterns dialog box.

| If *bwt* is | The border is |
|---|---|
| 1 | Hairline |
| 2 | Thin |
| 3 | Medium |
| 4 | Thick |

***Hlength*** is a number from 1 to 3 specifying the length of the arrowhead.

| If *hlength* is | The arrowhead is |
|---|---|
| 1 | Short |
| 2 | Medium |
| 3 | Long |

***Htype*** is a number from 1 to 3 specifying the style of the arrowhead.

| *Htype* | Style of arrowhead |
|---|---|
| 1 | No head |
| 2 | Open head |
| 3 | Closed head |

***Hwidth*** is a number from 1 to 3 specifying the width of the arrowhead.

| If *hwidth* is | The arrowhead is |
|---|---|
| 1 | Narrow |
| 2 | Medium |
| 3 | Wide |

***Invert*** is a logical value corresponding to the Invert If Negative check box in the Format Patterns dialog box. This argument applies only to data markers.

- If *invert* is TRUE, Microsoft Excel inverts the pattern in the selected item if it corresponds to a negative number.
- If *invert* is FALSE, Microsoft Excel removes the inverted pattern, if present, from the selected item corresponding to a negative value.

***Lauto*** is a number from 0 to 2 specifying line settings.



| If *lauto* is | Line settings are |
|---|---|
| 0 | Set by the user (custom) |
| 1 | Automatic (set by Microsoft Excel) |
| 2 | None |

***Lcolor*** is a number from 1 to 16 corresponding to the 16 line colors in the Format Patterns dialog box.

***Lstyle*** is a number from 1 to 8 corresponding to the eight line styles in the Format Patterns dialog box.

***Lwt*** is a number from 1 to 4 corresponding to the four line weights in the Format Patterns dialog box.

| If *lwt* is | The line is |
|---|---|
| 1 | Hairline |
| 2 | Thin |
| 3 | Medium |
| 4 | Thick |

***Mauto*** is a number from 0 to 2 specifying marker settings.

| If *mauto* is | Marker settings are |
|---|---|
| 0 | Set by the user |
| 1 | Automatic (set by Microsoft Excel) |
| 2 | None |

***Mback*** is a number from 1 to 16 corresponding to the 16 marker background colors in the Format Patterns dialog box.

***Mfore*** is a number from 1 to 16 corresponding to the 16 marker foreground colors in the Format Patterns dialog box.

***Mstyle*** is a number from 1 to 9 corresponding to the nine marker styles in the Format Patterns dialog box.

***Picture_units*** is the number of units you want each picture to represent in a scaled, stacked picture chart. This argument applies only to picture charts and only if *type* is 3.

***Rounded*** is a logical value corresponding to the Round Corners check box and specifying whether to make the corners on text boxes and rectangles rounded. If *rounded* is TRUE, the corners are rounded; if FALSE, the corners are square. If the selection is an arc or an oval, *rounded* is ignored.

***Shadow*** is a logical value corresponding to the Shadow check box. *Shadow* does not apply to area charts or bars in bar charts. If *shadow* is TRUE, Microsoft Excel adds a shadow to the selected item; if FALSE, Microsoft Excel removes the shadow, if one is present, from the selected item. If the selection is an arc, *shadow* is ignored.

***Tlabel*** is a number from 1 to 4 specifying the position of tick labels.

| *Tlabel* | Tick label position |
|---|---|
| 1 | None |
| 2 | Low |
| 3 | High |
| 4 | Next to axis |

***Tmajor*** is a number from 1 to 4 specifying the type of major tick marks.

| *Tmajor* | Type of major tick marks |
|---|---|
| 1 | None |
| 2 | Inside |
| 3 | Outside |
| 4 | Cross |

***Tminor*** is a number from 1 to 4 specifying the type of minor tick marks.

| *Tminor* | Type of minor tick marks |
|---|---|
| 1 | None |
| 2 | Inside |
| 3 | Outside |
| 4 | Cross |

***Type*** is a number from 1 to 3 specifying the type of pictures to use in a picture chart.

| *Type* | **Specifies that pictures should be** |
|---|---|
| 1 | Stretched to reach a particular value |
| 2 | Stacked on top of each other to reach a particular value |
| 3 | Stacked on top of each other, but you specify the number of units each picture represents |

### *Notes*

- You can select many graphic objects on a worksheet or macro sheet and apply formatting to them at the same time, but you can select only one chart item at a time.
- If you select multiple objects and if one or more of the objects requires a different form of the PATTERNS function, then choose the form corresponding to the object with the most formatting attributes—that is, the form with the most arguments. If you specify an argument that does not apply to an item, the argument has no effect on that item.
- To apply formatting to similar items on a chart, use the *apply* argument described above.

### *Example*

The following macro formula formats the selected text box with custom borders in border style 1 and border color 5, a shadow, and rounded corners. All other arguments are left in the default settings.

```
PATTERNS(0,1,5,,TRUE,,,,,TRUE)
```

### *Related Functions*

| | |
|---|---|
| FORMAT.FONT | Applies a font to the current selection |
| FORMAT.TEXT | Controls the alignment and orientation of the selected text |

## *PI()*

Returns the number 3.14159265358979, the mathematical constant π, accurate to 15 digits.

### *Examples*

PI()/2 equals 1.57079...

SIN(PI()/2) equals 1

If the radius of a circle is stored in a cell named Radius, the following formula calculates the area of the circle:

```
PI()*(Radius^2)
```

### *Related Functions*

| | |
|---|---|
| COS | Returns the cosine of a number |
| SIN | Returns the sine of a number |
| TAN | Returns the tangent of a number |

*Macro Sheets Only*

## *PLACEMENT(placement_type)*

Equivalent to the Format Object Placement command. Determines how the selected object or objects are attached to the cells beneath them. The way an object is attached to the cells beneath it affects how the object is moved or sized whenever you move or size the cells. For detailed information on the Format Object Placement command, see online Help.

***Placement_type*** is a number from 1 to 3 specifying how to attach the selected object or objects.

| **If *placement_type* is** | **The selected object is** |
|---|---|
| 1 | Moved and sized with cells. |
| 2 | Moved but not sized with cells. |
| 3 | Free-floating—it is not affected by moving and sizing cells. |

If an object is not selected, PLACEMENT returns the #VALUE! error value. If *placement_type* is omitted, the current status is unchanged.



### *Related Functions*

| | |
|---|---|
| CREATE.OBJECT | Creates the specified object |
| FORMAT.MOVE | Moves the selected object |
| FORMAT.SIZE | Changes the size of the selected object |

## *PMT(rate,nper,pv,fv,type)*

Returns the periodic payment for an annuity based on constant payments and a constant interest rate.

For a more complete description of the arguments in PMT, see PV later in this book.

***Rate*** is the interest rate per period.

***Nper*** is the total number of payment periods in an annuity.

***Pv*** is the present value—the total amount that a series of future payments is worth now.

***Fv*** is the future value, or a cash balance you want to attain after the last payment is made. If *fv* is omitted, it is assumed to be 0 (the future value of a loan, for example, is 0).

***Type*** is the number 0 or 1 and indicates when payments are due. Zero indicates payments are due at the end of the period; one indicates they are due at the beginning of the period. If *type* is omitted, it is assumed to be zero.

### *Note*

The payment returned by PMT includes principal and interest but no taxes, reserve payments, or fees sometimes associated with annuities.

### *Tip*

To find the total amount paid over the duration of the annuity, multiply the returned PMT value by *nper*.

### *Examples*

The following formula returns the monthly payment on a $10,000 loan at an annual rate of 8% that you must pay off in 10 months:

```
PMT(8%/12,10,10000) equals –$1,037.03
```

For the same loan, if payments are due at the beginning of the period, the payment is:

```
PMT(8%/12,10,10000,0,1) equals –$1,030.16
```

The following formula returns the amount someone must pay to you each month if you loan that person $5,000 at 12% and want to be paid back in five months:

```
PMT(12%/12,5,–5000) equals $1,030.20
```

Suppose you want to save $50,000 in 18 years by saving a constant amount each month. If you assume you'll be able to earn 6% interest on your savings, you can use PMT to determine how much to save each month:

```
PMT(6%/12,18*12,0,50000) equals –$129.08
```

If you pay $129.08 into a 6% savings account every month for 18 years, you will have $50,000.

### *Related Functions*

| | |
|---|---|
| FV | Returns the future value of an investment |
| IPMT | Returns the interest payment for an investment for a given period |
| NPER | Returns the number of payments for an investment |
| PPMT | Returns the payment on the principal for an investment for a given period |
| PV | Returns the present value of an investment |
| RATE | Returns the interest rate per period of an investment |

*Macro Sheets Only*

## *POKE(channel_num,item_text,data_ref)*

Sends data to another application. Use POKE to send data to documents in other applications you are communicating with through DDE.

*Important*

Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

***Channel_num*** is the channel number returned by a previously run INITIATE function.

***Item_text*** is text that identifies the application you are accessing through *channel_num*. The form of *item_text* depends on the application connected to *channel_num*.

***Data ref*** is a reference to the document containing the data to send.

If POKE is not successful, it returns the following values.

| Value returned | Meaning |
|---|---|
| #VALUE! | *Channel_num* is not a valid channel number. |
| #DIV/0! | The application you are accessing does not respond after a certain length of time, and you press ESC to cancel. |
| #REF! | POKE is refused. |

For information on accessing other applications, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*.

***Examples***

In Microsoft Excel for Windows, the following macro inserts the text from cell C3 into the Microsoft Word for Windows document SALES.DOC at the bookmark named COMM.BMK:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | SendData | |
| 4 | SendChanl | =INITIATE("WINWORD","C:\SALES.DOC") | |
| 5 | | =POKE(SendChanl,"COMM.BMK",C3) | Send comments to doc |
| 6 | | =TERMINATE(SendChanl) | |
| 7 | | =RETURN() | |

In Microsoft Excel and Microsoft Word for OS/2, the first argument to the INITIATE function would be PMWORD.

In Microsoft Excel for the Macintosh, the following macro inserts the text from cell C3 into the Microsoft Word document named Report:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | SendData | |
| 4 | SendChanl | =INITIATE("Word","HardDisk:Apps:Word:Report") | |
| 5 | | =POKE(SendChanl,"TopicName",C3) | Send comments to doc |
| 6 | | =TERMINATE(SendChanl) | |
| 7 | | =RETURN() | |

***Related Functions***

| | |
|---|---|
| INITIATE | Opens a channel to another application |
| REQUEST | Returns data from another application |
| TERMINATE | Closes a channel to another application |

## PPMT(*rate,per,nper,pv,fv,type*)

Returns the payment on the principal for a given period for an investment based on periodic, constant payments and a constant interest rate.

For a more complete description of the arguments in PPMT, see PV later in this manual.

***Rate*** is the interest rate per period.

***Per*** specifies the period and must be in the range 1 to *nper*.

***Nper*** is the total number of payment periods in an annuity.

***Pv*** is the present value—the total amount that a series of future payments is worth now.

***Fv*** is the future value, or a cash balance you want to attain after the last payment is made. If *fv* is omitted, it is assumed to be 0 (the future value of a loan, for example, is 0).

***Type*** is the number 0 or 1 and indicates when payments are due. Zero indicates payments are due at the end of the period; one indicates they are due at the beginning of the period. If *type* is omitted, it is assumed to be zero.



***Examples***

The following formula returns the principal payment for the first month of a two-year $2,000 loan at 10% annual interest:

PPMT(10%/12,1,24,2000) equals –$75.62

The following function returns the principal payment for the last year of a 10-year $200,000 loan at 8% annual interest:

PPMT(8%,10,10,200000) equals –$27,598.10

***Related Functions***

| | |
|---|---|
| FV | Returns the future value of an investment |
| IPMT | Returns the interest payment for an investment for a given period |
| NPER | Returns the number of payments for an investment |
| PMT | Returns the periodic total payment for an investment |
| PV | Returns the present value of an investment |
| RATE | Returns the interest rate per period of an investment |

*Macro Sheets Only*

## PRECISION(*logical*)

Equivalent to turning on or off the Precision As Displayed check box in the Options Calculation dialog box. Controls how values are stored in cells. Use PRECISION when the results of formulas do not seem to match the values used to calculate the formulas.

***Logical*** is a logical value corresponding to the Precision As Displayed check box in the Options Calculation dialog box.

- If *logical* is TRUE, Microsoft Excel turns on the check box and stores values exactly as they are displayed.
- If *logical* is FALSE, Microsoft Excel turns off the check box and stores future entries at full precision (15 digits).

*Warning*

The PRECISION function may permanently alter your data. PRECISION(TRUE) causes Microsoft Excel to change values on your worksheet or macro sheet to match displayed values.
PRECISION(FALSE) causes Microsoft Excel to store future values at full precision, but it does not restore previously entered numbers to their original values.

***Notes***

- Precision As Displayed does not affect numbers in General format. Numbers in General format are always calculated to full precision.
- Microsoft Excel calculates slightly faster when using full precision because with Precision As Displayed turned on, Microsoft Excel has to round off numbers as it calculates.

***Related Functions***

| | |
|---|---|
| FORMAT.NUMBER | Formats numbers, dates, and times |
| WORKSPACE | Changes the workspace settings for a document |

*Macro Sheets Only*

## PREFERRED( )

Equivalent to the Gallery Preferred command (full menus). Changes the format of the active chart to the format currently defined by the Gallery Set Preferred command or the SET.PREFERRED macro function.

***Related Function***

| | |
|---|---|
| SET.PREFERRED | Changes the default chart format |

Macro Sheets Only

## PRINT(range_num,from,to,copies,draft,preview,print_what,color,feed)
## PRINT?(range_num,from,to,copies,draft,preview,print_what,color,feed)

Equivalent to the File Print command. Prints the active document. For detailed information on the File Print command, see online Help.

Arguments correspond to options, check boxes, and edit boxes in the File Print dialog box. Arguments corresponding to check boxes are logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

**Range_num** is a number specifying which pages to print.

| Range_num | Prints the following pages |
|---|---|
| 1 | All the pages |
| 2 | Prints a specified range. If *range_num* is 2, then *from* and *to* are required arguments. |

**From** specifies the first page to print (ignored unless *range_num* equals 2).

**To** specifies the last page to print (ignored unless *range_num* equals 2).

**Copies** specifies the number of copies to print.

**Draft** corresponds to the Draft Quality check box.

**Preview** corresponds to the Preview check box.

**Print_what** is a number from 1 to 3 that specifies what parts of the worksheet or macro sheet to print. If a chart is the active document, *print_what* is ignored.

| Print_what | Prints |
|---|---|
| 1 | Document only |
| 2 | Notes only |
| 3 | Document and then notes |

**Color** corresponds to the Print Using Color check box. *Color* is available only in Microsoft Excel for the Macintosh.

**Feed** is a number specifying the type of paper feed. *Feed* is available only in Microsoft Excel for the Macintosh.

| Feed | Type of paper feed |
|---|---|
| 1 | Continuous (paper cassette) |
| 2 | Cut sheet or manual (manual feed) |

### Example

The following formula prints two copies of pages 5–7, including notes, in color.

```
PRINT(2,5,7,2,FALSE,FALSE,3,TRUE,)
```

### Related Functions

| | |
|---|---|
| PAGE.SETUP | Sets page printing specifications |
| PRINT.PREVIEW | Previews pages and page breaks before printing |
| PRINTER.SETUP | Identifies the printer and changes printer settings |
| SET.PRINT.AREA | Defines the print area |
| SET.PRINT.TITLES | Defines text to print as titles |

Macro Sheets Only

## PRINT.PREVIEW( )

Equivalent to the File Print Preview command. Previews the pages and page breaks of the active document on the screen before printing.

### Related Function

| | |
|---|---|
| PRINT | Prints the active document |



Macro Sheets Only

## PRINTER.SETUP(printer_text)
## PRINTER.SETUP?(printer_text)

Equivalent to the File Printer Setup command in Microsoft Excel for Windows or OS/2. Identifies the printer you want to use. Use PRINTER.SETUP to change the printer you are using.

**Printer_text** is the name of the printer to be activated. Enter *printer_text* exactly as it appears in the File Printer Setup dialog box.

### Note

This function is only available in Microsoft Excel for Windows or OS/2.

### Example

The following macro formula changes the printer to an HP LaserJet+ attached to COM1:

```
PRINTER.SETUP("HP LaserJet+ on COM1:")
```

### Related Functions

| | |
|---|---|
| PAGE.SETUP | Sets page printing specifications |
| PRINT | Prints the active document |

## PRODUCT(number1,number2,...)

Multiplies all the numbers given as arguments and returns the product.

**Number1, number2,...** are 1 to 14 numbers that you want to multiply.

- Arguments that are numbers, empty cells, logical values, or text representations of numbers are counted; arguments that are error values or text that cannot be translated into numbers cause errors.
- If an argument is an array or reference, only numbers in the array or reference are counted. Empty cells, logical values, text, or error values in the array or reference are ignored.

### Examples

If cells A2:C2 contain 5, 15, and 30:

```
PRODUCT(A2:C2) equals 2,250
PRODUCT(A2:C2,2) equals 4,500
```

### Related Functions

| | |
|---|---|
| FACT | Returns the factorial of a number |
| SUM | Adds its arguments |
| SUMPRODUCT | Returns the sum of the products of corresponding array components |

Macro Sheets Only

## PROMOTE(rowcol)
## PROMOTE?(rowcol)

Equivalent to clicking the promote button. Promotes the currently selected rows or columns in an outline. Use PROMOTE to change the configuration of an outline by promoting rows or columns of information.

**Rowcol** specifies whether to promote rows or columns.

| Rowcol | Demotes |
|---|---|
| 1 or omitted | Rows |
| 2 | Columns |

### Notes

- If the selection consists of an entire row or rows, then rows are promoted even if *rowcol* is 2. Similarly, selection of an entire column overrides *rowcol* 1.
- Also, if the selection is unambiguous (an entire row or column), then PROMOTE? will not display the dialog box.

### Related Functions

| | |
|---|---|
| DEMOTE | Demotes the selected rows or columns in an outline |
| SHOW.DETAIL | Expands or collapses a portion of an outline |
| SHOW.LEVELS | Displays a specific number of levels of an outline |

## PROPER(text)

Capitalizes the first letter in *text* and any other letters in *text* that follow any character other than a letter. Converts all other letters to lowercase.

**Text** is text enclosed in quotation marks, a formula that returns text, or a reference to a cell containing the text you want to partially capitalize.

### Examples

PROPER("this is a TITLE") equals "This Is A Title"

PROPER("2-cent's worth") equals "2-Cent'S Worth"

PROPER("76BudGet") equals "76Budget"

Suppose you keep information about vendors and parts in several similar worksheets. The following lines in a macro subroutine use the PROPER and UPPER functions to correct errors in capitalization that might occur when someone updates one of these worksheets using a custom dialog box. Assume that the custom dialog box is defined on another part of the macro sheet.

| | A | B | C |
|---|---|---|---|
| 20 | *Names* | *Formulas* | *Comments* |
| 21 | | | |
| 22 | | **CorrectDialogBox** | Corrects case |
| 23 | | =FORMULA(PROPER(DialogVendor),"Vendor") | These lines copy the |
| 24 | | =FORMULA(PROPER(DialogStreet),"Street") | corrected values from |
| 25 | | =FORMULA(PROPER(DialogCity),"City") | a dialog box area on |
| 26 | | =FORMULA(UPPER(DialogState),"State") | the macro sheet to six |
| 27 | | =FORMULA(DialogZip,"Zip") | named cells on the |
| 28 | | =FORMULA(DialogPart,"Part") | active worksheet. |
| 29 | | =RETURN() | |

The names Vendor, Street, City, State, Zip, and Part refer to specific cells on the external, active worksheet.

### Related Functions

| | |
|---|---|
| LOWER | Changes every letter in a text value to lowercase |
| UPPER | Capitalizes every letter in a text value |

*Macro Sheets Only*

## PROTECT.DOCUMENT(contents,windows, password,objects)

## PROTECT.DOCUMENT?(contents,windows, password,objects)

Equivalent to the Options Protect Document and Options Unprotect Document commands (full menus) if a worksheet or macro sheet is the active document. Equivalent to the Chart Protect Document and Chart Unprotect Document commands (full menus) if a chart is the active document. Adds protection to or removes protection from the active worksheet, macro sheet, or chart. Use PROTECT.DOCUMENT to prevent yourself or others from changing cell contents, objects, or window configurations in a document.

**Contents** is a logical value corresponding to the Cells check box in the Options Protect Document dialog box or the Chart check box in the Chart Protect Document dialog box.

- If *contents* is TRUE or omitted, Microsoft Excel turns on the check box and protects cells on the worksheet or macro sheet, or protects the entire chart (including objects) if the document is a chart.
- If *contents* is FALSE, Microsoft Excel turns off the check box (and removes protection if the correct password is supplied).

**Windows** is a logical value corresponding to the Windows check box in the Options Protect Document or the Chart Protect Document dialog box.



- If *windows* is TRUE, Microsoft Excel turns on the check box and prevents a document's windows from being moved or sized.
- If *windows* is FALSE or omitted, Microsoft Excel turns off the check box (and removes protection if the correct password is supplied).

**Password** is the password you specify in the form of text to protect or unprotect the file.

- If *password* is omitted when you protect a document, then you will be able to unprotect the document without a password. This is useful if you want only to protect the document from accidental changes.
- If *password* is omitted when you try to unprotect a protected document, the normal password dialog box is displayed.
- Passwords are not recorded into the PROTECT.DOCUMENT function when you use the macro recorder.
- If *password* is specified, Microsoft Excel still displays the password dialog box when you run the macro. You must re-enter the password to verify it.

**Objects** is a logical value corresponding to the Objects check box in the Options Protect Document dialog box. This argument applies only to worksheets and macro sheets.

- If *objects* is TRUE or omitted, Microsoft Excel turns on the check box and protects all locked objects on the worksheet or macro sheet.
- If *objects* is FALSE, Microsoft Excel turns off the check box.

#### Notes

- If *contents*, *windows*, and *objects* are FALSE, PROTECT.DOCUMENT carries out the Unprotect Document command. If *contents*, *windows*, or *objects* is TRUE, it carries out the Protect Document command.
- Make sure you protect macro sheets that protect or unprotect worksheets. If you type a password directly into the macro function either to protect or unprotect a worksheet (for example, PROTECT.DOCUMENT(TRUE,TRUE,"XD1411C",TRUE)), then someone could open the macro sheet and see the password needed to unprotect the worksheet.
- You can use the following commands, or their corresponding macro functions, on unprotected cells in a protected document: Edit Copy, Edit Paste, Edit Fill Right, Edit Fill Down, Formula Paste Name, and Formula Paste Function.

#### Warning

If you forget the password of a document that was previously protected with a password, then you cannot unprotect the document.

### Example

To protect the active document's contents, windows, and objects:

PROTECT.DOCUMENT(TRUE,TRUE,,TRUE)

### Related Functions

| | |
|---|---|
| CELL.PROTECTION | Protects a cell or range of cells on a worksheet |
| OBJECT.PROTECTION | Controls how an object is protected |
| SAVE.AS | Allows a password to be specified for a file to be opened |

## PV(rate,nper,pmt,fv,type)

Returns the present value of an investment. The present value is the total amount that a series of future payments is worth now. For example, when you borrow money, the loan amount is the present value to the lender.

**Rate** is the interest rate per period. For example, if you obtain an automobile loan at a 10% annual interest rate and make monthly payments, your interest rate per month is 10%/12, or 0.83%. You would enter 10%/12, or 0.83%, or 0.0083, into the formula as the *rate*.

**Nper** is the total number of payment periods in an annuity. For example, if you get a four-year car loan and make monthly payments, your loan has 4*12, or 48 periods. You would enter 48 into the formula for *nper*.

***Pmt*** is the payment made each period and cannot change over the life of the annuity. Typically, *pmt* includes principal and interest but no other fees or taxes. For example, the monthly payments on a $10,000, four-year car loan at 12% are $263.33. You would enter -263.33 into the formula as the *pmt*.

***Fv*** is the future value, or a cash balance you want to attain after the last payment is made. If *fv* is omitted, it is assumed to be 0 (the future value of a loan, for example, is 0). For example, if you want to save $50,000 to pay for a special project in 18 years, then $50,000 is the future value. You could then make a conservative guess at an interest rate and determine how much you must save each month.

***Type*** is the number 0 or 1 and indicates when payments are due. Zero indicates payments are due at the end of the period; one indicates they are due at the beginning of the period. If *type* is omitted, it is assumed to be zero.

### Notes

- Make sure that you are consistent about the units you use for specifying *rate* and *nper*. If you make monthly payments on a four-year loan at 12% annual interest, use 12%/12 for *rate* and 4*12 for *nper*. If you make annual payments on the same loan, use 12% for *rate* and 4 for *nper*.
- The following functions apply to annuities:

  FV
  IPMT
  NPER
  PMT
  PPMT
  PV
  RATE

  An annuity is a series of constant cash payments made over a continuous period. For example, a car loan or a mortgage is an annuity. For more information, see the description for each annuity function.
- In annuity functions, cash you pay out, such as a deposit to savings, is represented by a negative number; cash you receive, such as a dividend check, is represented by a positive number. For example, a $1,000 deposit to the bank would be represented by the argument -1000 if you are the depositor and by the argument 1000 if you are the bank.
- Microsoft Excel solves for one financial argument in terms of the others. If *rate* is not 0, then:

$$pv * (1 + rate)^{nper} + pmt(1 + rate * type) * \left(\frac{(1 + rate)^{nper} - 1}{rate}\right) + fv = 0$$

If *rate* is 0, then:

$$(pv + pmt) * (nper + fv) = 0$$

### Example

Suppose you're thinking of buying an insurance annuity that pays $500 at the end of every month for the next 20 years. The cost of the annuity is $60,000 and the money paid out will earn 8%. You want to determine whether this would be a good investment. Using the PV function you find that the present value of the annuity is:

PV(0.08/12,12*20,500,,0) equals -$59,777.15

The result is negative because it represents money that you would pay—an outgoing cash flow. The present value of the annuity ($59,777.15) is less than what you are asked to pay ($60,000). Therefore, you determine this would not be a good investment.

### Related Functions

| | |
|---|---|
| FV | Returns the future value of an investment |
| IPMT | Returns the interest payment for an investment for a given period |
| NPER | Returns the number of payments for an investment |
| PMT | Returns the periodic total payment for an investment |
| PPMT | Returns the payment on the principal for an investment for a given period |
| RATE | Returns the interest rate per period of an investment |



*Macro Sheets Only*

## QUIT( )

Equivalent to the File Exit command in Microsoft Excel for Windows or OS/2. Equivalent to the File Quit command in Microsoft Excel for the Macintosh. Quits Microsoft Excel and closes any open documents. If open documents have unsaved changes, Microsoft Excel displays a message asking if you want to save them. You can use QUIT in an Auto_Close macro to force Microsoft Excel to quit when a particular worksheet or macro sheet is closed.

### Important

QUIT will not ask whether you want to save changes if you have turned off error checking with an ERROR(FALSE) function.

### Related Function

| | |
|---|---|
| FILE.CLOSE | Closes the active file |

## RAND( )

Returns a random number greater than or equal to 0 and less than 1. The numbers are returned from a list of random numbers, and a new random number is returned every time the worksheet is calculated.

### Tips

- To generate a random integer that is greater than or equal to zero but less than *n*, use:

  INT(RAND()*n)
- To generate a random integer between *a* and *b* (inclusive), use:

  INT(RAND()*(b–a+1))+a
- If you want to use RAND to generate a random number but don't want the numbers to change every time the cell is calculated:

  1 Type **=RAND( )** in the formula bar.
  2 Choose Options Calculate Now.

  The formula you typed is changed to a random number.
  3 Press ENTER to enter the number.

### Examples

To generate a random number greater than or equal to 0 but less than 100:

RAND()*100

To generate a random integer between 15 and 92:

INT(RAND()*78)+15

If you need to generate several random numbers in specific ranges, you may find it helpful to write a custom function using the formula in the tips above. The following custom function generates random integers in the range you specify:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | RandBetween | |
| 4 | | =ARGUMENT("LowNumber",1) | Lower number |
| 5 | | =ARGUMENT("HighNumber",1) | Higher number |
| 6 | | LowNumber=INT(LowNumber) | Converts to an integer |
| 7 | | HighNumber=INT(HighNumber) | |
| 8 | RandInteger | =INT(RAND()*(HighNumber-LowNumber+1))+LowNumber | Computes a random integer |
| 9 | | =RETURN(RandInteger) | |

The complete formula in cell B8 is:

```
=INT(RAND()*(HighNumber–LowNumber+1))+LowNumber
```

If you want to make the preceding custom function a volatile function, see the example for VOLATILE.

### Note

You could also use the following formulas to replace the formulas in cells B6 and B7:

SET.NAME("LowNumber",INT(LowNumber))
SET.NAME("HighNumber",INT(HighNumber))

For more information, see SET.NAME later in this book.

## RATE(nper,pmt,pv,fv,type,guess)

Returns the interest rate per period for an annuity. RATE is calculated by iteration and can have zero or more solutions. If the successive results of RATE do not converge to within 0.0000001 after 20 iterations, RATE returns the #NUM! error value.

See PV earlier in this manual for a complete description of the arguments *nper*, *pmt*, *pv*, *fv*, and *type*.

***Nper*** is the total number of payment periods in an annuity.

***Pmt*** is the payment made each period and cannot change over the life of the annuity. Typically, *pmt* includes principal and interest but no other fees or taxes.

***Pv*** is the present value—the total amount that a series of future payments is worth now.

***Fv*** is the future value, or a cash balance you want to attain after the last payment is made. If *fv* is omitted, it is assumed to be 0 (the future value of a loan, for example, is 0).

***Type*** is the number 0 or 1 and indicates when payments are due. Zero indicates payments are due at the end of the period; one indicates they are due at the beginning of the period. If *type* is omitted, it is assumed to be zero.

***Guess*** is your guess for what the rate will be.

- If you omit *guess*, it is assumed to be 10%.
- If RATE does not converge, try different values for *guess*. RATE usually converges if *guess* is between 0 and 1.

### Example

To calculate the rate of a four-year $8,000 loan with monthly payments of $200:

RATE(48,–200,8000) equals 0.0077

This is the monthly rate, because the period is monthly. The annual rate is 0.0077*12, which equals 9.2%.

### Related Functions

| | |
|---|---|
| FV | Returns the future value of an investment |
| IPMT | Returns the interest payment for an investment for a given period |
| NPER | Returns the number of payments for an investment |
| PMT | Returns the periodic total payment for an investment |
| PPMT | Returns the payment on the principal for an investment for a given period |
| PV | Returns the present value of an investment |

*Macro Sheets Only*

## REFTEXT(reference,a1)

Converts a reference to an absolute reference in the form of text. Use REFTEXT when you need to manipulate references with text functions. After manipulating the reference text, you can convert it back into a normal reference by using TEXTREF.

***Reference*** is the reference you want to convert.

***A1*** is a logical value specifying A1 or R1C1 references.

- If *a1* is TRUE, REFTEXT returns an A1-style reference.
- If *a1* is FALSE or omitted, REFTEXT returns an R1C1-style reference.

### Examples

REFTEXT(C3,TRUE) equals "$C$3"

REFTEXT(B2:F2) equals "R2C2:R2C6"

Suppose that you have two worksheets named Worksheet1 and Worksheet2, and you want to be able to quickly move from a specified range on Worksheet1 to the same range on Worksheet2. For example, if cell B5 is active on Worksheet1, then the following subroutine selects cell B5 on Worksheet2.



| | A | B | C |
|---|---|---|---|
| 31 | *Names* | *Formulas* | *Comments* |
| 32 | | | |
| 33 | | SelectCorrespondingCell | |
| 34 | TempSheet | =REFTEXT(ACTIVE.CELL(),TRUE) | Gets active cell's reference |
| 35 | PermText | =SUBSTITUTE(TempSheet,"Worksheet1","W | Changes the text |
| 36 | | =FORMULA.GOTO(TEXTREF(PermText,TR | Converts PermText to ref |
| 37 | | =RETURN() | |

The complete formulas in cells B35 and B36 are:

```
=SUBSTITUTE(TempSheet,"Worksheet1","Worksheet2",1)
```

```
=FORMULA.GOTO(TEXTREF(PermText),TRUE)
```

FORMULA.GOTO is used instead of SELECT because it works with any open worksheet; SELECT works only with the active worksheet.

The following macro automatically inserts a series of formulas in the current horizontal selection on the active worksheet. The formulas calculate the average of the cells above the selection. This is similar to the auto-sum button on the tool bar except that the macro calculates averages, not sums. For example, if cells A3:E10 contained a block of numbers and if the current selection were A15:E15, the macro would place five AVERAGE formulas in A15:E15 (AVERAGE(A3:A10), AVERAGE(B3:B10), and so on).

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | AnyAutoAverage | |
| 4 | StartCell | =REFTEXT(SELECTION()) | Gets reference of range |
| 5 | | =SELECT.END(3) | Selects bottom cell |
| 6 | Cell1 | =RELREF(ACTIVE.CELL(),TEXTREF(StartCell)) | Gets reference of first cell |
| 7 | | =SELECT.END(3) | Selects top cell of range |
| 8 | Cell2 | =RELREF(ACTIVE.CELL(),TEXTREF(StartCell)) | Gets reference of last cell |
| 9 | | =FORMULA.FILL("=AVERAGE("&Cell1&":"&Cell2 | Inserts formula |
| 10 | | =SELECT(TEXTREF(StartCell)) | |
| 11 | | =RETURN() | |

The complete formula in cell B9 is:

```
=FORMULA.FILL("=AVERAGE("&Cell1&":"&Cell2&")",
TEXTREF(StartCell))
```

Also see the example for TEXTREF.

### Related Functions

| | |
|---|---|
| ABSREF | Returns the absolute reference of a cell |
| DEREF | Returns the values of cells in a reference |
| OFFSET | Returns a reference offset from a given reference |
| RELREF | Returns the relative reference of a cell |
| TEXTREF | Converts text to a reference |

*Macro Sheets Only*

## REGISTER(module_text,procedure_text, type_text,function_text,argument_text)

**Form 1: For Microsoft Excel for Windows or OS/2**

## REGISTER(file_text,resource_text, type_text,function_text,argument_text)

**Form 2: For Microsoft Excel for the Macintosh**

Makes the specified dynamic link library or code resource accessible to Microsoft Excel and returns the number identifying the code for use by CALL and UNREGISTER. You can also specify the name and arguments to use with the function you are registering.

### Note

Because Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different types of code resources, REGISTER takes a slightly different form when used in each operating environment.

***Module_text*** or ***file_text*** is quoted text specifying the name of the dynamic link library (DLL) that contains the procedure (in Microsoft Excel for Windows or OS/2) or the name of the file that contains the code resource (in Microsoft Excel for the Macintosh) you want to access.

***Procedure_text*** or ***resource_text*** is quoted text specifying the name of the procedure in the DLL (in Microsoft Excel for Windows or OS/2) or code resource (in Microsoft Excel for the Macintosh) you want to access.

***Type_text*** is quoted text specifying the data type of the return value and the number and data types of

the arguments. The first letter of *type_text* specifies the return value. The codes you use for *type_text* are described in detail in Appendix A, "Using the CALL and REGISTER Functions."

***Function_text*** is quoted text specifying the name of the function as you would type it on a worksheet or macro sheet.

***Argument_text*** is quoted text specifying the names of the arguments you want to appear in the Formula Paste Function dialog box.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula creates a function named BESSEL using the DLL named MATH.DLL, which contains a procedure named BESSEL that accepts two arguments:

```
REGISTER("MATH.DLL","BESSEL","BIB","BESSEL","Index, Argument")
```

In Microsoft Excel for the Macintosh, the following macro formula creates a function named BESSEL using the code resource named MATH, which contains a procedure named BESSEL that accepts two arguments:

```
REGISTER("MATH","BESSEL","BIB","BESSEL","Index, Argument")
```

Once you have calculated this REGISTER function, the BESSEL function is listed in the Formula Paste Function dialog box and you can enter the function directly on a worksheet or macro sheet, for example:

```
BESSEL(10,10)
```

### Tip

To use functions in a DLL or code resource directly on a worksheet without first registering them from a macro sheet, use the CALL function. CALL can be used with the arguments from the REGISTER function on a worksheet or macro sheet to place the DLL or code resource in memory and to send arguments to the function. For more information about this use of CALL, see CALL earlier in this book.

### Related Functions

| | |
|---|---|
| CALL | Calls a procedure in a dynamic link library |
| UNREGISTER | Removes a resource that was previously registered by the REGISTER function |

*Macro Sheets Only*

## RELREF(*reference,rel_to_ref*)

Returns the reference of a cell or cells relative to the upper-left cell of *rel_to_ref*. The reference is given as an R1C1-style relative reference in the form of text, such as "R[1]C[1]".

***Reference*** is the cell or cells to which you want to create a relative reference.

***Rel_to_ref*** is the cell from which you want to create the relative reference. If *rel_to_ref* is omitted, it is assumed to be the active cell.

### Examples

RELREF($A$1,$C$3) equals "R[−2]C[−2]"

RELREF($A$1:$E$5,$C$3:$G$7) equals "R[−2]C[−2]:R[2]C[2]"

RELREF($A$1:$E$5,$C$3) equals "R[−2]C[−2]:R[2]C[2]"

The following macro enters a formula to sum a specified range into a specified cell using RELREF.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | RangeSum | |
| 4 | | =SET.NAME("WkCell",INPUT("Formula?",8)) | Sets name for dest cell |
| 5 | DestRef | =RELREF(INPUT("Range?",8),WkCell) | Gets reference of sum range |
| 6 | | =FORMULA("=SUM("&DestRef&")",WkCell) | Puts formula in dest cell |
| 7 | | =RETURN() | |



### Related Functions

| | |
|---|---|
| ABSREF | Returns the absolute reference of a cell |
| DEREF | Returns the value of the cells in a reference |
| OFFSET | Returns a reference offset from a given reference |

*Macro Sheets Only*

## REMOVE.PAGE.BREAK()

Equivalent to the Options Remove Page Break command. Removes manual page breaks that you set with the SET.PAGE.BREAK function or the Options Set Page Break command. If the active cell is not below or to the right of a manual page break, REMOVE.PAGE.BREAK takes no action. If the entire document is selected, REMOVE.PAGE.BREAK removes all manual page breaks. REMOVE.PAGE.BREAK does not remove automatic page breaks.

### Related Function

| | |
|---|---|
| SET.PAGE.BREAK | Sets a manual page break |

*Macro Sheets Only*

## RENAME.COMMAND(*bar_num,menu, command,name_text*)

Changes the name of a built-in or custom menu command or the name of a menu. Use RENAME.COMMAND to change the name of a command on a menu, for example, when you create two custom commands that toggle on the menu. Examples of two built-in commands that toggle are the Full Menus and Short Menus commands.

***Bar_num*** can be either the number of one of the Microsoft Excel built-in menu bars or the number returned by a previously run ADD.BAR function. See ADD.COMMAND for a list of ID numbers for built-in menu bars.

***Menu*** can be either the name of a menu as text or the number of a menu.

***Command*** can be either the name of the command as text or the number of the command to be renamed (the first command on a menu is command 1). If *command* is 0, RENAME.COMMAND renames the menu instead of the command. Because other macros can change the position of custom menu commands, you should use the name of the command rather than a number whenever possible.

If the specified menu bar, menu, or command does not exist, RENAME.COMMAND returns the #VALUE! error value.

***Name_text*** is the new name for the command.

### Tip

To specify an access key for the new name, precede the character you want to use with an ampersand (&). The access key is indicated by an underline under one letter of a menu or command name. In Microsoft Excel for the Macintosh, you can use the Command Underline option in the Options Workspace dialog box to turn command underlining on or off.

For information on custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Example

To rename the Show Totals command as Hide Totals, and to make the letter "o" in Hide Totals an access key, use the following macro formula:

```
RENAME.COMMAND(1,"File","Show Totals","Hide T&otals")
```

### Related Functions

| | |
|---|---|
| ADD.COMMAND | Adds a command to a menu |
| CHECK.COMMAND | Adds or deletes a check mark from a command |
| DELETE.COMMAND | Deletes a command from a menu |
| ENABLE.COMMAND | Enables or disables a command |

## REPLACE(old_text,start_num,num_chars, new_text)

Replaces a specified number of characters of text. Use REPLACE to exchange one text string for another. This function is useful when you want to change every character or set of characters to another character or set of characters.

***Old_text*** is text in which you want to replace a specific number of characters.

***Start_num*** is the number of the character within *old_text* to begin replacing with *new_text*. The number of the first character in *old_text* is 1.

***Num_chars*** is the number of characters in *old_text*, starting at *start_num*, that you want to replace with *new_text*.

***New_text*** is the text with which you want to replace *num_chars* in *old_text*. All of *new_text* always replaces *num_chars*. For example, if you want to replace just two characters in *old_text* with *new_text* that contains ten characters, all ten characters of *new_text* replace the two characters.

### Examples

The following formula replaces the five characters after the fifth character in *old_text* with *new_text:*

REPLACE("abcdefghijk",6,5,"*") equals "abcde*k"

The sixth through tenth characters are all replaced by *.

The following formula replaces the last two digits of 1990 with 91:

REPLACE("1990",3,2,"91") equals "1991"

If cell A2 contains "123456", then:

REPLACE(A2,1,3,"@") equals "@456"

If the RIGHT function returns "ABCDEF", then:

REPLACE(RIGHT(A3,6),1,6,"*") equals "*"

### Related Functions

| | |
|---|---|
| FORMULA.REPLACE | Replaces text throughout a document |
| MID | Extracts a number of characters from text |
| SEARCH | Finds one text value within another (not case sensitive) |
| SUBSTITUTE | Replaces a group of characters with another group of characters |
| TRIM | Removes spaces from text |

Macro Sheets Only

## REPLACE.FONT(font_num,name_text, size_num,bold,italic,underline,strike,color,outline, shadow)

Replaces one of the four built-in fonts in Microsoft Excel for Windows version 2.1 or earlier with a new font and style. To change the font of the selected cell or range as part of a macro, use the FORMAT.FONT function instead.

***Font_num*** is a number from 1 to 4 and is equivalent to font numbers 1 to 4 in the Format Font dialog box of Microsoft Excel for Windows version 2.1 or earlier.

***Name_text*** is the name of the new font, as it appears in the dialog box. For example, "Courier" is a font name.

***Size_num*** is the size of the new font in points.

***Bold*** is a logical value that specifies whether the font is bold. If *bold* is TRUE, the font is bold.

***Italic*** is a logical value that specifies whether the font is italic. If *italic* is TRUE, the font is italic.

***Underline*** is a logical value that specifies whether the font is underlined. If *underline* is TRUE, the font is underlined.

***Strike*** is a logical value that specifies whether the new font is formatted as struck through. If *strike* is TRUE, the font is struck through.



***Color*** is a number from 1 to 16 corresponding to the 16 colors in the Format Font dialog box; if *color* is 0, the color is automatic.

***Outline*** is a logical value that specifies whether the font is outlined. If *outline* is TRUE, the font is outlined. *Outline* is only available with Microsoft Excel for the Macintosh.

***Shadow*** is a logical value that specifies whether the font is shadowed. If *shadow* is TRUE, the font is shadowed. *Shadow* is only available with Microsoft Excel for the Macintosh.

### Note

If you use REPLACE.FONT with *font_num* as 1, all characters in your current workspace's standard font will be changed to the font, size, and style specified by the rest of the arguments.

### Related Function

| | |
|---|---|
| FORMAT.FONT | Applies a font to the current selection |

## REPT(text,number_times)

Repeats text a given number of times. Use REPT to fill a cell with a number of instances of a text string.

***Text*** is the text you want to repeat.

***Number_times*** is a positive number specifying the number of times to repeat *text*. If *number_times* is 0, REPT returns "" (empty text). If *number_times* is not an integer, it is truncated. The result of the REPT function cannot be longer than 255 characters.

### Examples

REPT("-",3) equals "*-*-*-"

If A3 contains "Sales", then:

REPT($A$3,2.9) equals "SalesSales"

### Related Functions

| | |
|---|---|
| ALIGNMENT | Aligns or wraps text in cells |
| FILL.DOWN, FILL.LEFT, FILL.RIGHT, and FILL.UP | Fill a cell with a specified character |
| FORMULA.FILL | Fills a range with a formula |

Macro Sheets Only

## REQUEST(channel_num,item_text)

Requests an array of a specific type of information from an application with which you have a DDE link. Use REQUEST with other Microsoft Excel DDE functions to move information from another application into Microsoft Excel.

### Note

Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

***Channel_num*** is a number returned by a previously run INITIATE function. *Channel_num* refers to a channel through which Microsoft Excel communicates with another program.

***Item_text*** is a code indicating the type of information you want to request from another application. The form of *item_text* depends on the application connected to *channel_num*. For more information, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*.

REQUEST returns the data as an array. For example, suppose the remote data to be returned came from a worksheet that looked like this:

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 3 | | | |
| 2 | 4 | 5 | 6 | | | |

REQUEST would return that data as the following array:

{1,2,3;4,5,6}

If REQUEST is not successful, it returns the following error values.

| Value returned | Situation |
|---|---|
| #VALUE! | *Channel_num* is not a valid channel number. |
| #N/A | The application you are accessing is busy doing something else. |
| #DIV/0! | The application you are accessing does not respond after a certain length of time, or you have pressed ESC or COMMAND+PERIOD to cancel. |
| #REF! | The request is refused. |

### Example

In Microsoft Excel for Windows or OS/2, the following macro opens a channel to Word for Windows and returns the text specified by the bookmark named BMK1:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | GetParagraphs | |
| 4 | WChan | =INITIATE("WINWORD","C:\REPORT.DOC") | Initiates channel to Word |
| 5 | | =REQUEST(WChan,"BMK1") | Returns text in bookmark |
| 6 | | =TERMINATE(WChan) | Terminates channel to Word |
| 7 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| EXECUTE | Carries out commands in another application |
| INITIATE | Opens a channel to another application |
| POKE | Sends data to another application |
| SEND.KEYS | Sends keystrokes to another application |

*Macro Sheets Only*

## RESTART(*level_num*)

Removes a number of RETURN statements from the stack. When one macro calls another, the RETURN statement at the end of the second macro returns control to the calling macro. You can use the RESTART function to determine which macro regains control.

***Level_num*** is a number specifying the number of previous RETURN statements you want to be ignored. If *level_num* is omitted, the next RETURN statement will halt macro execution.

For example, if the currently running macro has two "ancestors," that is, was called by one macro that, in turn, was called by another macro, using RESTART(1) in the third macro returns control to the first calling macro when the RETURN statement is encountered. The RESTART(1) formula removes one level of RETURN statements from Microsoft Excel's memory so that the second macro is skipped.

### Note

RESTART is particularly useful if you frequently use macros to call other macros that in turn call other macros. Use RESTART in combination with IF statements to prevent macro execution from returning to macros that called, either directly or indirectly, the currently running macro.

*Macro Sheets Only*

## RESULT(*type_num*)

Specifies the type of data a custom function or subroutine returns. Use RESULT to make sure your custom function or subroutine returns a value of the correct data type.



***Type_num*** is a number specifying the data type.

| *Type_num* | Type of returned data |
|---|---|
| 1 | Number |
| 2 | Text |
| 4 | Logical |
| 8 | Reference |
| 16 | Error |
| 64 | Array |

- *Type_num* can be the sum of the numbers above to allow for more than one possible result type. For example, if *type_num* is 12, which equals 4 + 8, the result can be a logical or a reference value.
- If you omit *type_num*, it is assumed to be 7. Since 7 equals 1 + 2 + 4, the value returned can be a number (1), text (2), or logical value (4).

For more information on using RESULT in custom functions and subroutines, see Chapter 17, "Creating and Using Custom Worksheet Functions," in the *Microsoft Excel User's Guide*.

### Examples

The following custom function returns profit as a number (data type 1). This example is also included in the description of ARGUMENT.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | ProfitCalc | Calculates profit |
| 4 | | =RESULT(1) | Result will be a number |
| 5 | | =ARGUMENT("NumSold",1) | Defines the three arguments |
| 6 | | =ARGUMENT("Cost",1) | |
| 7 | | =ARGUMENT("Price",1) | |
| 8 | Profit | =NumSold*(Price–Cost) | Computes the profit |
| 9 | | =RETURN(Profit) | Returns the profit |

The following custom function returns an array (data type 64) of currency values. This macro assumes that the exchange rates (PoundsPerDollar, FrancsPerDollar, YenPerDollar, and MarksPerDollar) are defined and correctly named on the active worksheet.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | Currency | Converts dollars |
| 4 | | =RESULT(64) | Result will be an array |
| 5 | | =ARGUMENT("Money") | |
| 6 | MoneyRange | =Money*!PoundsPerDollar | These four cells |
| 7 | | =Money*!FrancsPerDollar | are named |
| 8 | | =Money*!YenPerDollar | "MoneyRange" |
| 9 | | =Money*!MarksPerDollar | |
| 10 | | =RETURN(MoneyRange) | |

The following subroutine macro returns the number of characters you want starting from any position in the specified file. This macro is analogous to the MID function, which returns a number of characters from a text string.

| | A | B | C |
|---|---|---|---|
| 31 | *Names* | *Formulas* | *Comments* |
| 32 | | | |
| 33 | | ReadCharsFromPosition | |
| 34 | | =RESULT(2) | Result will be text |
| 35 | | =ARGUMENT("FileName") | |
| 36 | | =ARGUMENT("PositionNum") | |
| 37 | | =ARGUMENT("NumChars") | |
| 38 | FileNumber | =FOPEN(FileName) | Obtains file number |
| 39 | | =FPOS(FileNumber,PositionNum) | Positions file as specified |
| 40 | Characters | =FREAD(FileNumber,NumChars) | Reads specified characters |
| 41 | | =FCLOSE(FileNumber) | Closes the file |
| 42 | | =RETURN(Characters) | |

In Microsoft Excel for Windows or OS/2, the following macro formula would call the preceding subroutine and return 10 characters starting from the fifth character in a file named C:\BUDGETS\91PLAN.TXT:

```
ReadCharsFromPosition("C:\BUDGETS\91PLAN.TXT",5,10)
```

In Microsoft Excel for the Macintosh, the following macro formula would call the preceding subroutine and return 10 characters starting from the fifth character in a file named Hard Disk:Budgets:Plan for 91 (text):

```
ReadCharsFromPosition("Hard Disk:Budgets:Plan for 91
(text)",5,10)
```

### Related Function

| | |
|---|---|
| ARGUMENT | Describes arguments to a macro |

Macro Sheets Only

## RETURN(value)

Ends the currently running macro. If the currently running macro is a subroutine macro that was called by another macro, control is returned to the calling macro. If the currently running macro is a custom function, control is returned to the formula that called the custom function. If the currently running macro is a command macro started by the user with the Macro Run command or a shortcut key or by clicking an object, control is returned to the user.

***Value*** specifies what to return. *Value* is usually a name or a reference.

- If the macro is a custom function or a subroutine, *value* specifies what value to return. However, not all subroutines return values; the last line in macros that do not return values is =RETURN().
- If the macro is a command macro run by the user, *value* should be omitted.

For more information, see Chapter 17, "Creating and Using Custom Worksheet Functions," in the *Microsoft Excel User's Guide*.

### Note

RETURN signals the end of a macro. Every macro must end with a RETURN or HALT function, but not every macro returns values.

### Example

For examples of custom functions and a subroutine macro that return values, see RESULT earlier in this manual.

### Related Functions

| | |
|---|---|
| BREAK | Interrupts a FOR–NEXT, WHILE–NEXT, or FOR.CELL–NEXT loop |
| HALT | Stops all macros from running |

## RIGHT(text,num_chars)

Returns the last (or rightmost) character or characters in a text string.

***Text*** is the text string containing the characters you want to extract.

***Num_chars*** specifies how many characters you want to extract.

- *Num_chars* must be greater than zero.
- If *num_chars* is greater than the length of *text*, RIGHT returns all of *text*.
- If *num_chars* is omitted, it is assumed to be 1.

### Examples

RIGHT("Sale Price",5) equals "Price"

RIGHT("Stock Number") equals "r"

RIGHT is similar to LEFT; for more examples, see LEFT earlier in this manual.

### Related Functions

| | |
|---|---|
| LEFT | Extracts the leftmost characters from a text value |
| MID | Extracts characters based on a starting position and number of characters |

## ROUND(number,num_digits)

Rounds a number to the specified number of digits.

***Number*** is the number you want to round.

***Num_digits*** specifies the number of digits to which you want to round *number*.

- If *num_digits* is greater than 0, then *number* is rounded to the specified number of decimal places.
- If *num_digits* is 0, then *number* is rounded to the nearest integer.
- If *num_digits* is less than 0, then *number* is rounded to the left of the decimal point.



### Examples

ROUND(2.15,1) equals 2.2

ROUND(2.149,1) equals 2.1

ROUND(–1.475,2) equals –1.48

ROUND(21.5,–1) equals 20

### Related Functions

| | |
|---|---|
| INT | Rounds a number down to the nearest integer |
| MOD | Gives the remainder from division |
| TRUNC | Truncates a number to an integer |

## ROW(reference)

Returns the row number of *reference*.

***Reference*** is the cell or range of cells for which you want the row number.

- If *reference* is omitted, it is assumed to be the reference of the cell in which the ROW function appears.
- If *reference* is a range of cells and if ROW is entered as a vertical array, ROW returns the row numbers of *reference* as a vertical array.
- *Reference* cannot refer to multiple areas.

### Examples

ROW(A3) equals 3

When entered as an array formula in three vertical cells:
ROW(A3:B5) equals {3;4;5}

If ROW is entered in C5, then:
ROW() equals ROW(C5) equals 5

The following macro formula tells you which row the active cell is in:
ROW(ACTIVE.CELL())

### Related Functions

| | |
|---|---|
| COLUMN | Returns the column number of a reference |
| ROWS | Returns the number of rows in a reference |

Macro Sheets Only

## ROW.HEIGHT(height_num,reference,standard_height,type_num)
## ROW.HEIGHT?(height_num,reference,standard_height,type_num)

Equivalent to the Format Row Height command. Changes the height of the rows in a reference. For detailed information on the Format Row Height command, see online Help.

***Height_num*** specifies how high you want the rows to be in units of the font corresponding to the Normal cell style. If *standard_height* is TRUE, *height_num* is ignored.

***Reference*** specifies the rows for which you want to change the height.

- If *reference* is omitted, the reference is assumed to be the current selection.
- If *reference* is specified, it must be either an external reference to the active worksheet, such as !$2:$4 or !Database, or an R1C1-style reference in the form of text or a name, such as "R1:R3", "R[–4]:R[–2]", or Database.
- If *reference* is a relative R1C1-style reference in the form of text, it is assumed to be relative to the active cell.

***Standard_height*** is a logical value that sets the row height to the standard height as determined by the height of the fonts in each row.

- If *standard_height* is TRUE, Microsoft Excel sets the row height to a standard height that may vary from row to row depending on the fonts used in each row, ignoring *height_num*.

- If *standard_height* is FALSE or omitted, Microsoft Excel sets the row height according to *height_num*.

***Type_num*** is a number from 1 to 3 corresponding to choosing the Hide or Unhide button in the Format Row Height dialog box or setting the selection to a best-fit height.

| *Type_num* | Action taken |
|---|---|
| 1 | Hides the row selection by setting the row height to 0 |
| 2 | Unhides the row selection by setting the row height to the value set before the selection was hidden |
| 3 | Sets the row selection to a best-fit height, which varies from row to row depending on how large the font is in any cell in each row or on how many lines of text are wrapped |

### Notes

- If any of the argument settings conflict, such as when *standard_height* is TRUE and *type_num* is 3, Microsoft Excel uses the *type_num* argument and ignores any arguments that conflict with *type_num*.
- If you are recording a macro while using a mouse, and you change row heights by dragging the row border, Microsoft Excel records the reference of the rows using R1C1-style references in the form of text. If Relative Record is on, Microsoft Excel uses R1C1-style relative references. If Absolute Record is on, Microsoft Excel uses R1C1-style absolute references.

### Examples

The following macro formula makes the current selection the standard height:

```
ROW.HEIGHT(,,TRUE)
```

The following macro formula hides row 1:

```
ROW.HEIGHT(,"R1",,1) or ROW.HEIGHT(0,"R1")
```

The following macro formula restores row 1 to a best-fit height:

```
ROW.HEIGHT(,"R1",,3)
```

### Related Function

| | |
|---|---|
| COLUMN.WIDTH | Sets the widths of the selected columns |

## ROWS(*array*)

Returns the number of rows in a reference or array.

***Array*** is an array, an array formula, or a reference to a range of cells for which you want the number of rows.

### Examples

```
ROWS(A1:C4) equals 4

ROWS({1,2,3;4,5,6}) equals 2
```

The following macro formula calculates the number of rows in the current selection:

```
ROWS(SELECTION())
```

The following macro formula calculates the number of rows in a particular area (specified by AreaNum) of the current multiple selection:

```
ROWS(INDEX(SELECTION(),,,AreaNum))
```

### Related Functions

| | |
|---|---|
| COLUMNS | Returns the number of columns in a reference |
| ROW | Returns the row number of a reference |

*Macro Sheets Only*

## RUN(*reference,step*)
## RUN?(*reference,step*)

Equivalent to the Macro Run command. Runs a macro. For detailed information on the Macro Run command, see online Help.



***Reference*** is a reference to the macro you want to run or the number 1 or 2 directing Microsoft Excel to run all Auto_Open or Auto_Close macros on the active macro sheet. If the macro sheet containing the macro is not the active document, *reference* can be an external reference to the name of the macro, such as RUN(MACRO.XLM!Months) or a quoted external R1C1-style reference to the location of the macro, such as RUN("MACRO.XLM!R2C3").

- If you use a reference to a range of cells as the *reference* argument, RUN begins with the macro function in the upper-left cell of *reference*.
- If *reference* is 1, Microsoft Excel runs all Auto_Open macros on the active macro sheet.
- If *reference* is 2, Microsoft Excel runs all Auto_Close macros on the active macro sheet.
- If *reference* is omitted, the macro function at the active cell is carried out, and macro execution continues down that column.

***Step*** is a logical value specifying that the macro is to be run in single-step mode. If step is TRUE, Microsoft Excel runs the macro in single-step mode; if FALSE or omitted, Microsoft Excel runs the macro normally.

### Notes

- RUN is recorded when you choose the Macro Run command while recording a macro. The reference you enter in the Macro Run dialog box is recorded as text, with A1-style references converted to R1C1-style references.
- To run a macro from a macro sheet, you could alternatively enter the name of the macro as a formula, followed by a set of parentheses. For example, enter =MACRO.XLM!Months() instead of =RUN(MACRO.XLM!Months).

### Examples

The following formula begins executing the macro commands at the active cell:

```
RUN()
```

In Microsoft Excel for Windows or OS/2, each of the following formulas runs the macro beginning at the upper-left corner of the range named Months, which begins at cell B14, on the macro sheet MACROS.XLM:

```
RUN(MACROS.XLM!Months)
RUN("MACROS.XLM!R14C2")
```

In Microsoft Excel for the Macintosh, each of the following formulas runs the macro beginning at the upper-left corner of the range named Months, which begins at cell B14, on the macro sheet Macros:

```
RUN(Macros!Months)
RUN("Macros!R14C2")
```

### Related Functions

| | |
|---|---|
| GOTO | Directs a macro to continue running at another reference |
| ON functions | Run a macro when a specified condition is TRUE |

*Macro Sheets Only*

## SAVE()

Equivalent to the File Save command. Saves the active document. For detailed information on the File Save command, see online Help.

### Note

Use the SAVE.AS function instead of SAVE when you want to change the name or file type, specify a password, create a backup file, or save a file to a different directory or folder.

### Related Functions

| | |
|---|---|
| SAVE.AS | Saves a document and allows you to specify the name, file type, password, backup file, and location of the document |
| SAVE.WORKSPACE | Saves a workspace file |

Macro Sheets Only

## SAVE.AS(*document_text,type_num,prot_pwd, backup,write_res_pwd,read_only_rec*)

## SAVE.AS?(*document_text,type_num,prot_pwd, backup,write_res_pwd,read_only_rec*)

Equivalent to the File Save As command. Use SAVE.AS to specify a new filename, file type, protection password, or write-reservation password, or to create a backup file. You can also use SAVE.AS to save a file as an add-in document or a template. For detailed information on the File Save As command, see online Help.

***Document_text*** specifies the name of a document to save, such as SALES.XLS (in Microsoft Excel for Windows or OS/2) or Sales East (in Microsoft Excel for the Macintosh). You can also include a full pathname in *document_text*, such as C:\EXCEL\ANALYZE.XLM (in Microsoft Excel for Windows or OS/2) or Hard Disk:Financials:East (in Microsoft Excel for the Macintosh).

***Type_num*** is a number from 1 to 26 specifying the file format in which to save the document. For information on file formats, see Chapter 3, "Creating and Using a Worksheet," in the *Microsoft Excel User's Guide*. The following table lists the file format corresponding to each *type_num*.

| *Type_num* | File format |
|---|---|
| 1 | Normal |
| 2 | SYLK |
| 3 | Text |
| 4 | WKS |
| 5 | WK1 |
| 6 | CSV |
| 7 | DBF2 |
| 8 | DBF3 |
| 9 | DIF |
| 10 | Reserved |
| 11 | DBF4 |
| 12 | Reserved |
| 13 | Reserved |
| 14 | Reserved |
| 15 | WK3 |
| 16 | Microsoft Excel 2.x |
| 17 | Template |
| 18 | Add-in macro |
| 19 | Text (Macintosh) |
| 20 | Text (Windows) |
| 21 | Text (OS/2 or DOS) |
| 22 | CSV (Macintosh) |
| 23 | CSV (Windows) |
| 24 | CSV (OS/2 or DOS) |
| 25 | International macro |
| 26 | International add-in macro |

The following table shows which values of *type_num* apply to the three Microsoft Excel document types.

| Document type | *Type_num* |
|---|---|
| Worksheet | All except 10, 12–14, 18, 25, 26 |
| Chart | 1, 16, 17 |
| Macro sheet | 1–3, 6, 9, 16–26 |

***Prot_pwd*** is a password given as quoted text or as a reference to a cell containing text. *Prot_pwd* should be no more than 15 characters. If a file is saved with a password, the password must be supplied for the file to be opened.

***Backup*** is a logical value that specifies whether to make a backup document. If *backup* is TRUE, Microsoft Excel creates a backup file; if FALSE, no backup file is created. If *backup* is omitted, the status is unchanged.

***Write_res_pwd*** is a password that allows the user to write to a file. If a file is saved with a password and the password is not supplied when the file is opened, the file is opened read only.



***Read_only_rec*** is a logical value specifying whether to save the document as read only recommended. If *read_only_rec* is TRUE, Microsoft Excel saves the document as read only recommended; if FALSE, Microsoft Excel saves the document normally. If *read_only_rec* is omitted, Microsoft Excel uses the current settings.

### Tip

To mark the active file as saved without writing it to a disk, set *type_num* equal to 0 and omit all other arguments—that is, SAVE.AS(,0). This is useful for closing a document without saving changes and preventing Microsoft Excel from displaying a message asking to save changes.

### Related Functions

| | |
|---|---|
| CLOSE | Closes the active window |
| GET.DOCUMENT | Returns information about a document |
| SAVE | Saves the active document |
| SAVE.WORKSPACE | Saves a workspace file |

Macro Sheets Only

## SAVE.NEW.OBJECT(*object_name*)

## SAVE.NEW.OBJECT?(*object_name*)

Equivalent to choosing the File Save New Object command when you're working with Microsoft Excel objects in NewWave. Saves a copy of the current Microsoft Excel object as a new object under the name you specify.

### Notes

- This function is only available if you are using Microsoft Excel with NewWave.
- In NewWave, Microsoft Excel objects correspond to directories.

***Object_name*** is text specifying the pathname for the new object. *Object_name* follows the usual DOS conventions. *Object_name* must be a new directory.

- If *object_name* is a pathname, then it is used without modifications.
- If *object_name* is only a short name that does not specify a complete path, then it is appended to the default Microsoft Excel directory (usually \HPNWDOS\EXCEL) to create a complete pathname.

### Note

In NewWave, SAVE.NEW.OBJECT replaces the SAVE.WORKSPACE function. If you use SAVE.WORKSPACE while running Microsoft Excel in NewWave, it will return the #VALUE! error value.

### Examples

The following macro function saves the object as the directory specified:

```
SAVE.NEW.OBJECT("\MYOBJECT\BUDGET")
```

The following macro function saves the object as \HPNWDOS\EXCEL\BUDGET:

```
SAVE.NEW.OBJECT("BUDGET")
```

### Related Functions

| | |
|---|---|
| ATTRIBUTES | Sets attributes for a Microsoft Excel object in NewWave |
| SHARE | Places a shared view of the current selection on the Clipboard in NewWave |
| SHARE.NAME | Places a shared view of a named range on the Clipboard in NewWave |

Macro Sheets Only

## SAVE.WORKSPACE(*name_text*)

## SAVE.WORKSPACE?(*name_text*)

Equivalent to the File Save Workspace command (full menus). Saves a workspace file. Workspace files include a list of the currently open documents, their position and status, and various workspace and display settings. The preferred chart format is also saved with a workspace. Use workspace files to open a set of related documents simultaneously and to save your workspace and display option settings. For detailed information on the File Save Workspace command, see online Help.

***Name_text*** specifies the name of a workspace document to save, such as SALES.XLW (in Microsoft Excel for Windows or OS/2) or Start Sales (in Microsoft Excel for the Macintosh). You can also include a drive specifier and pathname in *name_text*, such as C:\EXCEL\TRACKING.XLW or HardDisk:Excel:Tracking.

If *name_text* is omitted, it is assumed to be RESUME.XLW (in Microsoft Excel for Windows or OS/2) or Resume (in Microsoft Excel for the Macintosh) or the name of the last workspace document, if any, opened in the current session.

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula saves the current set of open documents as a workspace file named SALES.XLW:

```
SAVE.WORKSPACE("SALES")
```

In Microsoft Excel for the Macintosh, the following formula saves the current set of open documents as a workspace file named Sales Docs:

```
SAVE.WORKSPACE("Sales Docs")
```

### Related Functions

| | |
|---|---|
| OPEN | Opens an existing file or workspace |
| SAVE | Saves the active document |
| SAVE.AS | Saves a document and allows you to specify the name, file type, password, backup file, and location of the document |

Macro Sheets Only

## SCALE(*cross,cat_labels,cat_marks,between,max,reverse*)

## SCALE?(*cross,cat_labels,cat_marks,between,max,reverse*)

**Form 1: Category (x) axis, 2-D chart**

Equivalent to the Format Scale command. There are five forms of this function. This form of SCALE applies if the selected axis is a category (x) axis on a 2-D chart and the chart is not an xy (scatter) chart. Use the category-axis form of SCALE to change the position, formatting, and scaling of the category axis. For detailed information on the Format Scale command, see online Help.

Arguments correspond to text boxes and check boxes in the Format Scale dialog box. Arguments corresponding to check boxes are logical values. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Cross*** is a number corresponding to the Value (Y) Axis Crosses At Category box. The default is 1. *Cross* is ignored if *max* is set to TRUE.

***Cat_labels*** is a number corresponding to the Number Of Categories Between Tick Labels box. The default is 1.

***Cat_marks*** is a number corresponding to the Number Of Categories Between Tick Marks box. The default is 1.

***Between*** corresponds to the Value (Y) Axis Crosses Between Categories check box. This argument only applies if *cat_labels* is set to a number other than 1.

***Max*** corresponds to the Value (Y) Axis Crosses At Maximum Category check box. If *max* is TRUE, it overrides any setting for *cross*.

***Reverse*** corresponds to the Categories In Reverse Order check box.

### Related Functions

| | |
|---|---|
| AXES | Controls whether axes on a chart are visible |
| GRIDLINES | Controls whether major and minor gridlines are visible |

Macro Sheets Only

## SCALE(*min_num,max_num,major,minor,cross,logarithmic,reverse,max*)

## SCALE?(*min_num,max_num,major,minor,cross,logarithmic,reverse,max*)

**Form 2: Value (y) axis on a 2-D chart, or x or y (value) axis on an xy (scatter) chart**

Equivalent to the Format Scale command. There are five forms of this function. This form of SCALE applies if the selected axis is a value (y) axis on a 2-D chart, or either axis on an xy (scatter) chart. Use the value-axis form of SCALE to change the position, formatting, and scaling of the value axis. For detailed information on the Format Scale command, see online Help.



The first five arguments correspond to the five range variables in the Format Scale dialog box. Each argument can be either the logical value TRUE or a number:

- If an argument is TRUE, Microsoft Excel turns on the Auto check box.
- If a number, that number is used for the variable.

***Min_num*** corresponds to the Minimum box and is the minimum value for the value axis.

***Max_num*** corresponds to the Maximum box and is the maximum value for the value axis.

***Major*** corresponds to the Major Unit box and is the major unit of measure.

***Minor*** corresponds to the Minor Unit box and is the minor unit of measure.

***Cross*** corresponds to the Category (X) Axis Crosses At box for the value (y) axis of a 2-D chart or the Value (Y) Axis Crosses At box for the x axis of an xy (scatter) chart.

The last three arguments are logical values corresponding to check boxes in the Format Scale dialog box. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Logarithmic*** corresponds to the Logarithmic Scale check box.

***Reverse*** corresponds to the Values/Categories In Reverse Order check box.

***Max*** corresponds to the Category/Value Axis Crosses At Maximum Value/Category check box.

Macro Sheets Only

## SCALE(*cat_labels,cat_marks,reverse,between*)

## SCALE?(*cat_labels,cat_marks,reverse,between*)

**Form 3: Category (x) axis, 3-D chart**

Equivalent to the Format Scale command. There are five forms of this function. This form of SCALE applies if the selected axis is a category (x) axis on a 3-D chart. Use the category axis form of SCALE to change the position, formatting, and scaling of the category axis. For detailed information on the Format Scale command, see online Help.

***Cat_labels*** is a number corresponding to the Number Of Categories Between Tick Labels box. The default is 1.

***Cat_marks*** is a number corresponding to the Number Of Categories Between Tick Marks box. The default is 1.

***Reverse*** corresponds to the Categories In Reverse Order check box in the Format Scale dialog box. If *reverse* is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Between*** corresponds to the Value (Z) Axis Crosses Between Categories check box. If *between* is TRUE, Microsoft Excel turns on the check box and the data points appear between categories. If *between* is FALSE or omitted, Microsoft Excel turns off the check box.

Macro Sheets Only

## SCALE(*series_labels,series_marks,reverse*)

## SCALE?(*series_labels,series_marks,reverse*)

**Form 4: Series (y) axis, 3-D chart**

Equivalent to the Format Scale command. There are five forms of this function. This form of SCALE applies if the selected axis is a series (y) axis on a 3-D chart. Use the series axis form of SCALE to change the position, formatting, and scaling of the series axis. For detailed information on the Format Scale command, see online Help.

***Series_labels*** is a number corresponding to the Number Of Series Between Tick Labels box. The default is 1.

***Series_marks*** is a number corresponding to the Number Of Series Between Tick Marks box. The default is 1.

***Reverse*** is a logical value that corresponds to the Series In Reverse Order check box in the Format Scale dialog box. If *reverse* is TRUE, Microsoft Excel displays the series in reverse order; if FALSE or omitted, Microsoft Excel displays the series normally.

*Macro Sheets Only*

## SCALE(*min_num,max_num,major,minor,cross, logarithmic,reverse,min*)

## SCALE?(*min_num,max_num,major,minor,cross, logarithmic,reverse,min*)

**Form 5: Value (z) axis, 3-D chart**

Equivalent to the Format Scale command. There are five forms of this function. This form of SCALE applies if the selected axis is a value (z) axis on a 3-D chart. Use the value axis form of SCALE to change the position, formatting, and scaling of the value axis. For detailed information on the Format Scale command, see online Help.

The first five arguments correspond to the five range variables in the Format Scale dialog box, as listed below. Each argument can be either the logical value TRUE or a number.

- If TRUE or omitted, the Auto check box is turned on.
- If a number, that number is used for the variable.

***Min_num*** corresponds to the Minimum box, and is the minimum value for the value axis.

***Max_num*** corresponds to the Maximum box, and is the maximum value for the value axis.

***Major*** corresponds to the Major Unit box, and is the major unit of measure.

***Minor*** corresponds to the Minor Unit box and is the minor unit of measure.

***Cross*** corresponds to the Floor (XY Plane) Crosses At: box.

The last three arguments are logical values corresponding to check boxes. If an argument is TRUE, Microsoft Excel turns on the check box; if FALSE, Microsoft Excel turns off the check box.

***Logarithmic*** corresponds to the Logarithmic Scale check box.

***Reverse*** corresponds to the Values In Reverse Order check box.

***Min*** corresponds to the Floor (XY Plane) Crosses At Minimum Value check box.

## SEARCH(*find_text,within_text,start_num*)

Returns the number of the character at which a specific character or text string is first found, reading from left to right. Use SEARCH to discover the location of a character or text string within another text string, so that you can use the MID or REPLACE functions to change the text.

***Find_text*** is the text you want to find. You can use the wildcard characters, question mark (?) and asterisk (*), in *find_text*. A question mark matches any single character; an asterisk matches any sequence of characters. If you want to find an actual question mark or asterisk, type a tilde (~) before the character. If *find_text* is not found, the #VALUE! error value is returned.

***Within_text*** is the text in which you want to search for *find_text*.

***Start_num*** is the character number in *within_text*, counting from the left, at which you want to start searching.

- If *start_num* is omitted, it is assumed to be 1.
- If *start_num* is not greater than 0 or is greater than the length of *within_text*, the #VALUE! error value is returned.



### *Tip*

Use *start_num* to skip a specified number of characters from the left of the text. For example, suppose you are working with a text string such as "AYF0093.YoungMensApparel". To find the number of the first "Y" in the descriptive part of the text string, set *start_num* equal to 8 so that the serial-number portion of the text is not searched. SEARCH begins with character 8, finds *find_text* at the next character, and returns the number 9. SEARCH always returns the number of characters from the left of the text string, not from *start_num*.

### *Notes*

- SEARCH does not distinguish between uppercase and lowercase letters when searching text.
- SEARCH is similar to FIND, except that FIND is case sensitive.

### *Examples*

SEARCH("e","Statements",1) equals 5

If cell B17 contains the word "margin" and cell A14 contains "Profit Margin", then:

SEARCH($B$17,$A$14) equals 8

Use SEARCH with the REPLACE function to provide REPLACE with the correct *start_num* at which to begin inserting new text. Using the same cell references as the previous example:

REPLACE($A$14,SEARCH($B$17,$A$14),6,"Amount") returns the text "Profit Amount".

### *Related Functions*

| | |
|---|---|
| FIND | Finds one text value within another (case sensitive) |
| MID | Extracts a number of characters from text |
| REPLACE | Replaces characters in a string of text |
| SUBSTITUTE | Replaces a group of characters with another group of characters |

## SECOND(*serial_number*)

Returns the second corresponding to *serial_number*. The second is given as an integer in the range 0 to 59. Use SECOND to get the time in seconds indicated by a serial number.

***Serial_number*** is the date-time code used by Microsoft Excel for date and time calculations. You can give *serial_number* as text, such as "16:48:23" or "4:48:47 PM", instead of as a number. The text is automatically converted to a serial number. For more information on *serial_number*, see NOW.

### *Note*

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

### *Examples*

SECOND("4:48:18 PM") equals 18

SECOND(0.01) equals 24

SECOND(4.02) equals 48

### *Related Functions*

| | |
|---|---|
| NOW | Returns the serial number of the current date and time |
| YEAR, MONTH, DAY, WEEKDAY, HOUR, and MINUTE | Convert serial numbers into years, months, days, days of the week, hours, and minutes |

*Macro Sheets Only*

## SELECT(*selection,active_cell*)

**Form 1: Cells on a worksheet or macro sheet**

Equivalent to selecting cells or changing the active cell. There are three forms of SELECT. Use this form to select a cell on a worksheet or macro sheet; use one of the other forms to select worksheet or macro sheet objects or chart items.

### Tip

You can enter data in a cell without selecting the cell by using the *reference* argument to the FORMULA function to refer to the cell in which you want to enter data. This method is usually considerably faster than using the SELECT function.

***Selection*** is the cell or range of cells you want to select. *Selection* can be a reference to the active worksheet, such as !$A$1:$A$3 or !Sales, or a quoted R1C1-style reference to a cell or range relative to the active cell in the current selection, such as "R[−1]C[−1]:R[1]C[1]". If *selection* is omitted, the current selection is used.

***Active_cell*** is the cell in *selection* you want to make the active cell. *Active_cell* can be a reference to a single cell on the active worksheet, such as !$A$1, or a quoted R1C1-style reference relative to the active cell, such as "R[−1]C[−1]". If *active_cell* is omitted, SELECT makes the cell in the upper-left corner of *selection* the active cell.

### Notes

- *Active_cell* must be within *selection*. If it is not, an error message is displayed and SELECT returns the #VALUE! error value.
- If you are recording a macro using the Macro Relative Record command (full menus) and you make a selection, Microsoft Excel records the action using R1C1-style relative references in the form of text.
- If you are recording using the Macro Absolute Record command (full menus), Microsoft Excel records the action using absolute references.
- You cannot give an external reference to a specific sheet as the *selection* argument. The sheet on which you want to make a selection must be active when you use SELECT. Use FORMULA.GOTO to make a selection on an external worksheet or macro sheet.

### Examples

The following macro formula selects cells C3:E5 on the active worksheet and makes C5 the active cell:

```
SELECT(!$C$3:$E$5,!$C$5)
```

If the active cell is C3, the following macro formula selects cells E5:G7 and makes cell F6 the active cell in the selection:

```
SELECT("R[2]C[2]:R[4]C[4]","R[1]C[1]")
```

You can also make multiple noncontiguous selections with SELECT. The following macro formula selects a number of noncontiguous ranges:

```
SELECT("R1C1,R3C2:R4C3,R8C4:R10C5")
```

The following sequence of macro formulas moves the active cell right, left, down, and up within the selection, just as TAB, SHIFT+TAB, ENTER, and SHIFT+ENTER do:

```
SELECT(,"RC[1]")
SELECT(,"RC[-1]")
SELECT(,"R[1]C")
SELECT(,"R[-1]C")
```

Use SELECT with the OFFSET function to select a new range a specified distance away from the current range. For example, the following macro formula selects a range that is the same size as the current range, one row down:

```
SELECT(OFFSET(SELECTION(),1,0))
```

### Related Functions

| | |
|---|---|
| ACTIVE.CELL | Returns the reference of the active cell |
| SELECT.SPECIAL | Selects a group of cells belonging to a category |
| SELECTION | Returns the reference of the current selection |

*Macro Sheets Only*

## SELECT(*object_id_text,replace*)

**Form 2: Worksheet and macro sheet objects**

Equivalent to selecting objects on a worksheet or macro sheet. There are three forms of SELECT. Use this form to select an object on which to perform an action; use one of the other forms to select cells on a worksheet or macro sheet or items on a chart.



***Object_id_text*** is text that identifies the object to select. *Object_id_text* can be the name of more than one object. To give the name of more than one object, use the following format:

```
SELECT("Oval 3, Arc 2, Line 4")
```

The last item in the *object_id_text* list will be the active object. The active object is important when moving and sizing a group of objects. A multiple selection of objects is moved and sized relative to the upper-left corner of the active object.

***Replace*** is a logical value that specifies whether previously selected objects are included in the selection. If *replace* is TRUE or omitted, Microsoft Excel only selects the objects specified by *object_id_text*; if FALSE, it includes any objects that were previously selected. For example, if a button is selected and a SELECT formula selects an arc and an oval, TRUE leaves only the arc and oval selected and FALSE includes the button with the arc and oval.

### Note

Objects can be identified by their object type and number as described in CREATE.OBJECT, or by the unique number that specifies the order of their creation. For example, if the third object you create is an oval, you could use either "oval 3" or "3" as *object_id_text*.

### Example

The following macro formulas each select a number of objects and specify Arc 2 as the active object:

```
SELECT("Oval 3, Arc 1, Line 4, Arc 2")
SELECT("3, 1, 4, 2")
```

### Related Functions

| | |
|---|---|
| FORMAT.MOVE | Moves the selected objects |
| FORMAT.SIZE | Changes the size of the selected objects |
| GET.OBJECT | Returns information about the selected objects |
| SELECTION | Returns the reference of the current selection |

*Macro Sheets Only*

## SELECT(*item_text,single_point*)

**Form 3: Chart Items**

Selects a chart object as specified by the selection code *item_text*. There are three forms of SELECT. Use this form to select a chart item to which you want to apply formatting; use one of the other forms to select cells or objects on a worksheet or macro sheet.

***Item_text*** is a selection code from the following table which specifies which chart object to select.

| To select | *Item_text* |
|---|---|
| Entire chart | "Chart" |
| Plot area | "Plot" |
| Legend | "Legend" |
| Main chart value axis | "Axis 1" |
| Main chart category axis | "Axis 2" |
| Overlay chart value axis or 3-D series axis | "Axis 3" |
| Overlay chart category axis | "Axis 4" |
| Chart title | "Title" |
| Label for the main chart value axis | "Text Axis 1" |
| Label for the main chart category axis | "Text Axis 2" |
| Label for the main chart series axis | "Text Axis 3" |
| *n*th floating text item | "Text *n*" |
| *n*th arrow | "Arrow *n*" |
| Major gridlines of value axis | "Gridline 1" |
| Minor gridlines of value axis | "Gridline 2" |
| Major gridlines of category axis | "Gridline 3" |
| Minor gridlines of category axis | "Gridline 4" |
| Major gridlines of series axis | "Gridline 5" |
| Minor gridlines of series axis | "Gridline 6" |
| Main chart droplines | "Dropline 1" |
| Overlay chart droplines | "Dropline 2" |

| To select | *Item_text* |
|---|---|
| Main chart hi-lo lines | "Hiloline 1" |
| Overlay chart hi-lo lines | "Hiloline 2" |
| Entire series | "S*n*" |
| Data associated with point *m* in series *n* if *single_point* is TRUE | "S*n*P*m*" |
| Text attached to point *m* of series *n* | "Text S*n*P*m*" |
| Series title text of series *n* of an area chart | "Text S*n*" |
| Base of a 3-D chart | "Floor" |
| Back of a 3-D chart | "Walls" |

***Single_point*** is a logical value that determines whether to select a single point. *Single_point* is available only when *item_text* is "S*n*P*m*". If *single_point* is TRUE, Microsoft Excel selects a single point; if FALSE or omitted, Microsoft Excel selects a single point if there is only one series in the chart or selects the entire series if there is more than one series in the chart. If you specify *single_point* when *item_text* is any value other than "S*n*P*m*", SELECT returns an error.

### Examples

SELECT("Chart") selects the entire chart.

SELECT("Dropline 2") selects the droplines of an overlay chart.

SELECT("S1P3",TRUE) selects the third point in the first series.

SELECT("Text S1") selects the series title text of the first series in an area chart.

Macro Sheets Only

## SELECT.CHART( )

Equivalent to the Chart Select Chart command. This function is equivalent to using the third form of SELECT with "Chart" as the *item_text* argument.

### Note

This function is included for compatibility with macros written with Microsoft Excel for the Macintosh version 1.5 or earlier.

Macro Sheets Only

## SELECT.END(*direction_num*)

Selects the cell at the edge of the range in the direction specified. Equivalent to pressing CTRL+ARROW key in Microsoft Excel for Windows or OS/2 or COMMAND+ARROW key in Microsoft Excel for the Macintosh.

***Direction_num*** is a number from 1 to 4 indicating the direction to move.

| *Direction_num* | Direction |
|---|---|
| 1 | Left (equivalent to CTRL+LEFT ARROW or COMMAND+LEFT ARROW) |
| 2 | Right (equivalent to CTRL+RIGHT ARROW or COMMAND+RIGHT ARROW) |
| 3 | Up (equivalent to CTRL+UP ARROW or COMMAND+UP ARROW) |
| 4 | Down (equivalent to CTRL+DOWN ARROW or COMMAND+DOWN ARROW) |

### Example

The following macro selects the cell at the bottom of the current vertical range of numbers and enters a SUM formula in a cell two rows below the bottom of the range:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | Selector | |
| 4 | SelRows | =ROWS(SELECTION()) | Number of rows in selection |
| 5 | | =SELECT.END(4) | Selects last cell in selection |
| 6 | BotRow | =ROW(ACTIVE.CELL()) | Gets the row number |
| 7 | | =FORMULA("=SUM(R"&BotRow&"C:R"&Bot | Enters SUM formula |
| 8 | | =RETURN() | |



The complete formula in cell B7 is:

```
=FORMULA("=SUM(R"&BotRow&"C:R"&BotRow-
SelRows&"C)","R[2]C")
```

### Related Function

SELECT.LAST.CELL — Selects the last cell on a worksheet or macro sheet

Macro Sheets Only

## SELECT.LAST.CELL( )

Equivalent to choosing the Formula Select Special command and selecting the Last Cell option. Selects the cell at the intersection of the last row and column that contains a formula, value, or format, or that is referred to in a formula or name. For detailed information on the Formula Select Special command, see online Help.

### Related Function

SELECT.END — Selects the last cell in a range

Macro Sheets Only

## SELECT.PLOT.AREA( )

Equivalent to the Chart Select Plot Area command. Selects the plot area of the active chart.

### Note

SELECT.PLOT.AREA is included only for compatibility with previous versions of Microsoft Excel for the Macintosh. SELECT.PLOT.AREA is the same as the SELECT("Plot") function.

### Related Function

SELECT — Selects cells or objects on a sheet or an item on a chart

Macro Sheets Only

## SELECT.SPECIAL(*type_num,value_type, levels*)

## SELECT.SPECIAL?(*type_num,value_type, levels*)

Equivalent to the Formula Select Special command (full menus). Use SELECT.SPECIAL to select groups of similar cells in one of a variety of categories. For detailed information on the Formula Select Special command, see online Help.

***Type_num*** is a number from 1 to 13 corresponding to options in the Formula Select Special dialog box and describes what to select.

| *Type_num* | Description |
|---|---|
| 1 | Notes |
| 2 | Constants |
| 3 | Formulas |
| 4 | Blanks |
| 5 | Current region |
| 6 | Current array |
| 7 | Row differences |
| 8 | Column differences |
| 9 | Precedents |
| 10 | Dependents |
| 11 | Last cell |
| 12 | Visible cells only (outlining) |
| 13 | All objects |

***Value_type*** is a number specifying which types of constants or formulas you want to select. *Value_type* is available only when *type_num* is 2 or 3.

| *Value_type* | Selects |
|---|---|
| 1 | Numbers |
| 2 | Text |
| 4 | Logical values |
| 16 | Error values |

These values can be added to select more than one type. The default for *value_type* is 23, select all value types.

***Levels*** is a number specifying how precedents and dependents are selected. *Levels* is available only when *type_num* is 9 or 10. The default is 1.

| *Levels* | Selects |
|---|---|
| 1 | Direct only |
| 2 | All levels |

*Macro Sheets Only*

## SELECTION( )

Returns the reference or object identifier of the selection as an external reference. Use SELECTION to return information about the current selection for use in other macro formulas. If an object is selected, SELECTION returns the identifier of the object. If multiple objects are selected, it returns the identifiers of all the selected objects, as a string separated by commas. If more than 255 characters would be returned, SELECTION returns the #VALUE! error value.

| Object type | Identifier returned |
|---|---|
| Imported graphic | Picture *n* |
| Linked graphic | Picture *n* |
| Chart picture | Picture *n* |
| Linked chart | Chart *n* |
| Worksheet range | Picture *n* |
| Linked worksheet range | Picture *n* |
| Text box | Text *n* |
| Button | Button *n* |
| Rectangle | Rectangle *n* |
| Oval | Oval *n* |
| Line | Line *n* |
| Arc | Arc *n* |
| Group | Group *n* |

### Note

If you use the value returned by SELECTION in a function or operation, you usually get the value contained in the selection instead of its reference. References are automatically converted to the contents of the reference. If you want to work with the actual reference, use the REFTEXT function to convert the reference to text, which you can then store or manipulate. See the example in TEXTREF.

### Examples

If the document in the active window is named Sheet1, and if A1:A3 is the selection, then:

SELECTION() equals Sheet1!A1:A3

The following macro formula moves the current selection one row down:

```
SELECT(OFFSET(SELECTION(),1,0))
```

The above formula is particularly useful for moving incrementally through a database to add or modify records.

The following macro formula defines a name to refer to one row below the current selection on the active worksheet:

```
DEFINE.NAME("EntryRange",OFFSET(SELECTION(),1,0))
```

The following macro formula places the object names in the current multiple selection as the name Objects.

```
SET.NAME("Objects",SELECTION())
```

### Related Functions

| | |
|---|---|
| ACTIVE.CELL | Returns the reference of the active cell |
| OFFSET | Returns a reference offset from a given reference |
| SELECT | Selects cells or objects on a sheet or an item on a chart |

*Macro Sheets Only*

## SEND.KEYS(key_text,wait_logical)

Sends keystrokes to the active application just as if they were typed at the keyboard. Use SEND.KEYS to send keystrokes that perform actions and execute commands to applications you are running with Microsoft Excel's other DDE functions.



### Note

This function is available only in Microsoft Excel for Windows or OS/2.

***Key_text*** is the key or key combination you want to send to another application. The format for *key_text* is described in the ON.KEY function.

***Wait_logical*** is a logical value that determines whether the macro continues before the actions caused by *key_text* are carried out.

- If *wait_logical* is TRUE, Microsoft Excel waits for the keys to be processed before returning control to the macro.
- If *wait_logical* is FALSE or omitted, the macro continues running without waiting for the keys to be processed.

### Note

If Microsoft Excel is the active application, *wait_logical* is assumed to be FALSE, even if you enter *wait_logical* as TRUE. This is because if *wait_logical* is TRUE, Microsoft Excel waits for the keys to be processed in the other application before returning control to the macro. Microsoft Excel doesn't process keys while a macro is running.

### Example

The following macro uses the Calculator application to multiply some numbers, and then cuts the result and pastes it into Microsoft Excel:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **CalcPaste** | |
| 4 | | =EXEC("CALC.EXE") | Starts the Calculator |
| 5 | | =SEND.KEYS("10*30",TRUE) | Sends 10*30 |
| 6 | | =SEND.KEYS("~",TRUE) | Sends ENTER |
| 7 | | =SEND.KEYS("%ec",TRUE) | Sends Edit Copy |
| 8 | | =APP.ACTIVATE(,FALSE) | Activates Microsoft Excel |
| 9 | | =SELECT(B1) | Selects cell B1 |
| 10 | | =PASTE() | Pastes result |
| 11 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| APP.ACTIVATE | Activates an application |
| EXECUTE | Carries out commands in another application |
| ON.KEY | Runs a macro when a specified key is pressed |

*Macro Sheets Only*

## SEND.MAIL(recipient,subject,return_receipt)
## SEND.MAIL?(recipient,subject,return_receipt)

Equivalent to the File Send Mail command. Sends the active document using Microsoft Mail. For detailed information on the File Send Mail command, see online Help.

### Note

This function is only available if you are using Microsoft Excel for the Macintosh and Microsoft Mail version 2.0 or later.

***Recipient*** is the name of the person to whom you want to send the mail. The name should be given as quoted text. To specify more than one name, give the list of names as an array. For example, SEND.MAIL({"John","Paul","George","Ringo"}) would send the active document to the four names in the array. You can also refer to a range on a worksheet or macro sheet that contains a list of names to whom you want the mail to be sent.

To send mail to users on different Microsoft Mail servers, specify the server name along with the user name. The following text, as the *recipient* argument, sends mail to waynegr on server2, gregpr on the current server, and victorge on server7:

```
{"waynegr@server2","gregpr",
"victorge@server7"}
```

***Subject*** is a quoted text string that specifies the subject of the message. If *subject* is omitted, the name of the active document is used as the subject.

***Return_receipt*** is a logical value that corresponds to the Return Receipt check box. If *return_receipt* is

TRUE, Microsoft Excel turns on the check box; if FALSE or omitted, Microsoft Excel turns off the check box.

### Examples

The following macro formula sends the active document or workgroup to John Peters, with the subject indicated and the Return Receipt check box turned on:

```
SEND.MAIL("John Peters","Last Quarter's Results",TRUE)
```

The following macro formula sends the active document or workgroup to the four people indicated, with the subject indicated and the Return Receipt check box turned off:

```
SEND.MAIL({"John Peters","Mary Hopkins","Bill
Perks","Stephanie Kurtz"},"WeeklyReport",FALSE)
```

Note that the list of names is surrounded by brackets and that commas separate the quoted names.

The following macro formula sends the active document or workgroup to the list of names found in the range named DistribList on the active macro sheet:

```
SEND.MAIL(DistribList,"Friday's Meeting",FALSE)
```

### Related Function

| | |
|---|---|
| OPEN.MAIL | Opens mail containing a document Microsoft Excel can open |

*Macro Sheets Only*

## SEND.TO.BACK( )

Equivalent to the Format Send To Back command. Sends the selected object or objects to the back. Use SEND.TO.BACK to send selected objects behind other objects. For detailed information on the Format Send To Back command, see online Help.

If the selection is not an object or a group of objects, SEND.TO.BACK returns the #VALUE! error value and interrupts the macro.

### Related Function

| | |
|---|---|
| BRING.TO.FRONT | Brings selected objects to the front |

*Charts Only*

## SERIES(*name_ref,categories,values,plot_order*)

Represents a data series in the active chart. SERIES is used only in charts; you cannot enter it on a worksheet or macro sheet. You normally create or change data series by using the EDIT.SERIES macro function, which is equivalent to the Chart Edit Series command. However, you can edit a data series manually by selecting it, activating the formula bar, and typing the changes.

***Name_ref*** is the name of the data series. It can be an external reference to a single cell or a name defined as a single cell. *Name_ref* can also be text enclosed in quotation marks (for example, "Projected Sales").

***Categories*** is an external reference to the name of the worksheet and to the cells that contain one of the following sets of data:

- Category labels for all charts except xy (scatter) charts
- X-coordinate data for xy (scatter) charts

***Values*** is an external reference to the name of the worksheet and to the cells that contain values (or y-coordinate data in scatter charts).

***Plot_order*** is an integer specifying whether the series is plotted first, second, or third, and so on, in the chart. No two series can have the same *plot_order*.

### Notes

- *Categories* and *values* can be arrays or references to a multiple selection, although they cannot be names that refer to a multiple selection. If you specify a multiple selection for any of these arguments, make sure you include the necessary sets of parentheses so that Microsoft Excel does not treat the components of the references as separate arguments.
- If either *categories* or *values* is a multiple selection, then all areas in that selection must be either vertical (more rows than columns) or horizontal (more columns than rows).



For more information on the SERIES function, see EDIT.SERIES in this book and Chapter 12, "Editing a Chart," in the *Microsoft Excel User's Guide*.

### Related Function

| | |
|---|---|
| EDIT.SERIES | Creates or changes a chart series |

*Macro Sheets Only*

## SET.CRITERIA( )

Equivalent to the Data Set Criteria command. Defines the name Criteria for the selected range on a worksheet or macro sheet. For detailed information on the Data Set Criteria command, see online Help.

*Macro Sheets Only*

## SET.DATABASE( )

Equivalent to the Data Set Database command. Defines the name Database for the selected range on a worksheet or macro sheet. For detailed information on the Data Set Database command, see online Help.

*Macro Sheets Only*

## SET.EXTRACT( )

Equivalent to the Data Set Extract command. Defines the name Extract for the selected range on the active sheet. For detailed information on the Data Set Extract command, see online Help.

*Macro Sheets Only*

## SET.NAME(*name_text,value*)

Defines a name on a macro sheet to refer to a value. The SET.NAME function is useful for storing values while the macro is calculating.

***Name_text*** is the name in the form of text that refers to *value*.

***Value*** is the value you want to store in *name_text*.

- If *value* is omitted, the name *name_text* is deleted.
- If *value* is a reference, *name_text* is defined to refer to that reference.

### Notes

- If you want to define a name as a constant value, you can use the following syntax instead of the SET.NAME function:

  *name_text=value*

  See the first two examples below.
- SET.NAME defines names as absolute references, even if a relative reference is specified. See the third and fourth examples below.
- If you want *name_text* to refer permanently to the value of a referenced cell rather than to the reference itself, you must use the DEREF function. Use of DEREF prevents *name_text* from referring to a new value every time the contents of the referenced cell changes. See the last example below.

### Tips

- If you need to return an array to a macro sheet (for example, if the macro needs a list of all open windows), assign a name to the array instead of placing the array information in a range of cells. For example:

  ```
  SET.NAME("OpenDocuments",WINDOWS()) or
  SET.NAME("OpenDocuments",{"Worksheet1",
  "Worksheet2"})
  ```
- When you're debugging a macro and want to know the current value assigned to a name created by SET.NAME, you can halt the macro, choose Formula Define Name, and select the name from the Formula Define Name dialog box.

### Examples

Each of these formulas defines the name Counter to refer to the constant number 1 on the macro sheet:

```
SET.NAME("Counter",1)
Counter=1
```

Each of these formulas redefines Counter to refer to the current value of Counter plus 1:

```
SET.NAME("Counter",Counter+1)
Counter=Counter+1
```

The following macro formula defines the name Reference to refer to cell $A$1:

```
SET.NAME("Reference",A1)
```

The following macro formula defines the name Results to refer to the cells $A$1:$C$3:

```
SET.NAME("Results",A1:C3)
```

The following macro formula defines the name Range as the current selection:

```
SET.NAME("Range",SELECTION())
```

If $A$1 contains the value 2, the following macro formula defines the name Index to refer to the constant value 2:

```
SET.NAME("Index",DEREF(A1))
```

For more examples, see NAMES and FILES earlier in this manual.

### Related Functions

| | |
|---|---|
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| SET.VALUE | Sets the value of a cell on a macro sheet |

Macro Sheets Only

## SET.PAGE.BREAK()

Equivalent to the Options Set Page Break command (full menus). Sets manual page breaks for a printed worksheet. Use SET.PAGE.BREAK to override the automatic page breaks. Setting a manual page break changes the automatic page breaks that follow it.

The page break occurs above and to the left of the active cell and appears as dotted lines if you have set up a printer. If the active cell is in column A, a manual page break is added only above the cell. If the active cell is in row 1, a manual page break is added only at the left edge of the cell. If the row or column next to the active cell already has a page break, SET.PAGE.BREAK takes no action.

### Related Functions

| | |
|---|---|
| PRINT.PREVIEW | Previews pages and page breaks before printing |
| REMOVE.PAGE.BREAK | Removes manual page breaks |

Macro Sheets Only

## SET.PREFERRED()

Equivalent to the Gallery Set Preferred command (full menus). Changes the default format that Microsoft Excel uses when you create a new chart or when you format a chart with the Gallery Preferred command or the PREFERRED macro function. When you use the SET.PREFERRED function, the format of the active chart becomes the preferred format.

### Related Function

| | |
|---|---|
| PREFERRED | Changes the format of the active chart to the preferred format |

Macro Sheets Only

## SET.PRINT.AREA()

Equivalent to the Options Set Print Area command. Defines the print area—the area that prints when you choose the File Print command or use the PRINT macro function.

### Note

If you use SET.PRINT.AREA with a multiple selection and then use the PRINT function, the individual selections are printed one after the other in the order they were selected. To print the entire worksheet again, select the entire worksheet and use SET.PRINT.AREA. You can also delete the name Print_Area by choosing the Delete button in the Formula Define Name dialog box or by running the DELETE.NAME macro function.



### Example

The following macro prints the database on the active external worksheet and then deletes the name Print_Area:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **PrintDatabase** | |
| 4 | | =SELECT(!Database) | |
| 5 | | =SET.PRINT.AREA() | |
| 6 | | =PRINT() | |
| 7 | | =DELETE.NAME("Print_Area") | |
| 8 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| DELETE.NAME | Deletes a name |
| PRINT | Prints the active document |
| SET.PRINT.TITLES | Identifies text to print as titles |

Macro Sheets Only

## SET.PRINT.TITLES()

Equivalent to the Options Set Print Titles command (full menus). Uses the current selection to define the text you want as titles in the printed document. Use SET.PRINT.TITLES if you want Microsoft Excel to print the titles whenever it prints any cells in a row or column that intersects the print titles area; a cell need only share the row or column with a print title for the title to be printed above or to the left of that cell.

### Notes

- The print titles selection must consist of the entire row and/or columns containing the text you want to use for titles and can be a multiple selection. Microsoft Excel names this selection Print_Titles when SET.PRINT.TITLES is run.
- Do not include the print titles as part of the print area, or the titles will be printed twice on the first page.

### Related Functions

| | |
|---|---|
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| PRINT | Prints the active document |
| SET.PRINT.AREA | Identifies text to print as titles |

Macro Sheets Only

## SET.UPDATE.STATUS(*link_text,status, type_of_link*)

Sets the update status of a link to automatic or manual. Use SET.UPDATE.STATUS to change the way a link is updated.

***Link_text*** is the pathname of the linked file for which you want to change the update status.

***Status*** is the number 1 or 2 and describes how you want the link to be updated.

| *Status* | Update method |
|---|---|
| 1 | Automatic |
| 2 | Manual |

***Type_of_link*** is a number from 1 to 4 that specifies what type of link you want to get information about.

| *Type_of_link* | Link document type |
|---|---|
| 1 | Not available |
| 2 | DDE link |
| 3 | Not available |
| 4 | Outgoing NewWave link |

### Example

In Microsoft Excel for Windows or OS/2, the following macro formula sets the update status of the DDE link to Microsoft Word for Windows to manual:

```
SET.UPDATE.STATUS("WinWord\Memo|DDE-LINK",2,2)
```

### Related Functions

| | |
|---|---|
| GET.LINK.INFO | Returns information about a link |
| UPDATE.LINK | Updates a link to another document |

*Macro Sheets Only*

## SET.VALUE(reference,values)

Changes the value of a cell or cells on the macro sheet without changing any formulas entered in those cells. Use SET.VALUE to assign initial values and to store values during the calculation of a macro. SET.VALUE is especially useful for initializing a dialog box and the conditional test in a WHILE loop. SET.VALUE assigns values to a specific reference or to the name of a reference that has already been defined. Use SET.NAME to create a new name and to assign a new value to the name.

***Reference*** specifies the cell or cells on the macro sheet to which you want to assign a new value or values.

- If a cell in *reference* previously contained a formula, the formula is not changed, but the value of the cell might change. See the second example below.
- If *reference* is a reference to a range of cells, rather than to a single cell, then *values* should be an array of the same size. If not, Microsoft Excel expands it into multiple *values* using the normal rules for expanding arrays. See the third example below.

***Values*** is the value or set of values you want to assign to the cell or cells in *reference*.

### Notes

- Use FORMULA to change values in a worksheet cell.
- Use SET.NAME to create names on a macro sheet.

### Examples

The following macro formula changes the value of cell A1 on the macro sheet to 1:

```
SET.VALUE($A$1,1)
```

Suppose the name TempAverage refers to a cell containing the formula AVERAGE(Temp1,Temp2, Temp3). The following formula assigns the value 99 to this cell, even if the average of the arguments is not 99, without changing the formula in TempAverage:

```
SET.VALUE(TempAverage,99)
```

The preceding formula is useful if a WHILE loop or some other conditional test depends on TempAverage and you want to force the conditional test to have a particular result. Of course, TempAverage is restored to its correct value as soon as it is recalculated. (Recall that unlike formulas in a worksheet, formulas in a macro sheet are not recalculated until the macro actually uses them.)

The following macro formula stores the values 1, 2, 3, and 4 in cells A1:B2:

```
SET.VALUE($A$1:$B$2,{1,2;3,4})
```

The following macro uses WHILE and FOR loops to generate prime numbers. As the macro generates more and more primes, it gradually slows down, and it ends when it can't find any more primes in the time you specify. SET.VALUE initializes variables when the macro starts.



| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | PrimeNumbers | Returns prime numbers |
| 4 | StopSeconds | =INPUT("Stop after (seconds)?",1) | How long to look? |
| 5 | | =SET.VALUE(Continue,TRUE) | |
| 6 | | =SET.VALUE(NextNum,3) | Starts testing at 3 |
| 7 | | =SET.VALUE(CurrentTime,NOW()) | Initializes time marker |
| 8 | | =WHILE(Continue=TRUE) | Keeps testing more primes |
| 9 | | =SET.NAME("TestFailed",FALSE) | Initialize TestFailed variable |
| 10 | | =FOR("Divisor",3,INT(NextNum/2),2) | Odd number loop |
| 11 | Quotient | =NextNum/Divisor | |
| 12 | | =IF((Quotient-TRUNC(Quotient))=0) | If no remainder, then |
| 13 | | Divisor=NextNum-1 | not prime; adjusts loop |
| 14 | | TestFailed=TRUE | counter to cancel FOR |
| 15 | | =END.IF() | loop; the test failed |
| 16 | | =NEXT() | |
| 17 | | =IF(TestFailed,GOTO(NextNum)) | If failed, tests NextNum |
| 18 | Continue | =(SECOND(NOW()-CurrentTime))<StopSec | Time too long? |
| 19 | CurrentTime | =NOW() | |
| 20 | | =FORMULA(NextNum) | Places prime in worksheet |
| 21 | | =SELECT(OFFSET(ACTIVE.CELL(),1,0)) | Moves to next row |
| 22 | NextNum | =NextNum+2 | Tests next odd number |
| 23 | | =NEXT() | |
| 24 | | =RETURN() | |

The complete formula in cell B18 is:

```
=(SECOND(NOW()-CurrentTime))<StopSeconds
```

### Related Functions

| | |
|---|---|
| DEFINE.NAME | Defines a name to refer to a reference, value, or formula |
| FORMULA | Enters a formula in a cell or text on a chart |
| SET.NAME | Defines a name on a macro sheet |

*Macro Sheets Only*

## SHARE()

Equivalent to choosing the Edit Share command when you're working with Microsoft Excel objects in NewWave. Places a shared view of the selection on the Clipboard.

### Notes

- This function is only available if you are using Microsoft Excel with NewWave.
- SHARE returns the #VALUE! error value if the selection is a multiple selection.

### Related Functions

| | |
|---|---|
| ATTRIBUTES | Sets attributes for a Microsoft Excel object in NewWave |
| GET.LINK.INFO | Returns information about a link |
| SAVE.NEW.OBJECT | Saves a copy of the Microsoft Excel object as a new object in NewWave |
| SET.UPDATE.STATUS | Sets the update status of a link to automatic or manual |
| SHARE.NAME | Places a shared view of a named range on the Clipboard in NewWave |
| UPDATE.LINK | Updates a link to another document |

*Macro Sheets Only*

## SHARE.NAME(range_name)

Equivalent to choosing the Edit Share command when you're working with Microsoft Excel objects in NewWave. Places a shared view of the range you specify on the Clipboard.

### Note

This function is only available if you are using Microsoft Excel with NewWave.

***Range_name*** is the name of the range that you want to share on the Clipboard.

### Example

If the range A1:D4 is defined as January Grades, then the following macro formula shares the region A1:D4:

```
SHARE.NAME(January Grades)
```

Macro Sheets Only

## SHORT.MENUS(logical)

Equivalent to the Options Short Menus command (full menus) or the Options Full Menus command if a worksheet or macro sheet is the active document. Equivalent to the Chart Short Menus command (full menus) or the Chart Full Menus command if a chart is the active document. Use SHORT.MENUS in a macro to set up the working environment with short or long menus depending on whether the user will be using Microsoft Excel's advanced features.

***Logical*** is a logical value specifying whether to use short or full menus. If *logical* is TRUE or omitted, Microsoft Excel displays short menus; if FALSE, Microsoft Excel displays full menus.

Macro Sheets Only

## SHOW.ACTIVE.CELL()

Equivalent to the Formula Show Active Cell command or to pressing CTRL+BACKSPACE in Microsoft Excel for Windows or OS/2 or to pressing COMMAND+DELETE in Microsoft Excel for the Macintosh. Scrolls the active window so the active cell becomes visible. If an object is selected, SHOW.ACTIVE.CELL returns the #VALUE! error value.

### Tip

You can use IS functions to test whether an object or cell is selected. ISERROR(ACTIVE.CELL()) returns TRUE if an object is selected, and ISREF(ACTIVE.CELL()) returns TRUE if a cell is selected.

### Related Functions

| | |
|---|---|
| ACTIVE.CELL | Returns the reference of the active cell |
| FORMULA.GOTO | Selects the specified reference or named range |

Macro Sheets Only

## SHOW.BAR(bar_num)

Displays the specified menu bar. Use SHOW.BAR to display a menu bar you have created with the ADD.BAR function or to display a built-in Microsoft Excel menu bar.

***Bar_num*** is the number of the menu bar you want to display. It can be the number of one of the Microsoft Excel built-in menu bars, the number returned by a previously executed ADD.BAR function, or a reference to a cell containing a previously executed ADD.BAR function. If *bar_num* is omitted, Microsoft Excel displays the appropriate menu bar for the active document.

| If active window contains | Bar displayed |
|---|---|
| A worksheet or macro sheet (full menus) | 1 |
| A chart (full menus) | 2 |
| No active window | 3 |
| The Info window | 4 |
| A worksheet or macro sheet (short menus) | 5 |
| A chart (short menus) | 6 |

### Notes

- When displaying a built-in menu bar, you can display only bars 1 or 5 if a worksheet or macro sheet is active, bars 2 or 6 if a chart is active, and so on. If you try to display a chart menu bar while a worksheet or macro sheet is active, SHOW.BAR returns an error and interrupts the current macro.
- Displaying a custom menu bar disables automatic menu-bar switching when different types of documents are selected. For example, if a custom menu bar is displayed and you activate a chart, neither of the two chart menus is automatically displayed as it would be when you are using the built-in menu bars. Automatic menu-bar switching is reenabled when a built-in bar is displayed using SHOW.BAR.



For information on custom menus, see Chapter 20, "Creating a Custom Application," in the *Microsoft Excel User's Guide*.

### Example

The following macro creates a new menu bar, adds menus to the new menu bar and a command to the built-in menu bar, and displays the new menu bar:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | NewMenuBar | |
| 4 | NewBar | =ADD.BAR() | Creates a new menu bar |
| 5 | | =ADD.MENU(NewBar,NewFile) | Adds NewFile menu |
| 6 | | =ADD.MENU(NewBar,NewEdit) | Adds NewEdit menu |
| 7 | | =ADD.MENU(NewBar,NewReport) | Adds NewReport menu |
| 8 | | =ADD.MENU(NewBar,NewFormat) | Adds NewFormat menu |
| 9 | | =ADD.MENU(NewBar,NewOptions) | Adds NewOptions menu |
| 10 | | =ADD.COMMAND(1,"File",ShowNewBar) | Adds command to File menu |
| 11 | | =SHOW.BAR(NewBar) | Displays NewBar |
| 12 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| ADD.BAR | Creates a new menu bar |
| DELETE.BAR | Deletes a menu bar |

Macro Sheets Only

## SHOW.CLIPBOARD()

Equivalent to the Window Show Clipboard command in Microsoft Excel for OS/2 or Microsoft Excel for the Macintosh. Equivalent to running the Clipboard application from the Control menu in Microsoft Excel for Windows. Displays the contents of the Clipboard in a new window. For detailed information on the Window Show Clipboard command, see online Help.

### Notes

- In Microsoft Excel for Windows, the Clipboard must already be available on the desktop if you want to display its contents in a new window. If it is not already available, you must run the SHOW.CLIPBOARD function twice—once to make the Clipboard application available and again to display it in a new window.
- To close the Clipboard window, use the CLOSE function.
- If the Clipboard contains cells, the window shows the size of the Clipboard contents in rows and columns. If the Clipboard contains text cut from the formula bar, the window displays the text.

### Example

The following macro asks if you want to see the Clipboard before you paste it.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | ShowBeforePasting | |
| 4 | ViewBefore | =ALERT("View before pasting?",1) | |
| 5 | | =IF(ViewBefore) | If OK, shows Clipboard |
| 6 | | = SHOW.CLIPBOARD() | |
| 7 | Paste | = ALERT("Paste Clipboard?",1) | Asks if you want to paste |
| 8 | | = IF(NOT(Paste)) | If you don't paste, then |
| 9 | | = CLOSE() | closes Clipboard |
| 10 | | = RETURN() | and returns |
| 11 | | = ELSE() | If you do paste, then |
| 12 | | = CLOSE() | closes Clipboard first |
| 13 | | = END.IF() | |
| 14 | | =END.IF() | |
| 15 | | =PASTE() | Pastes Clipboard contents |
| 16 | | =RETURN() | |

Macro Sheets Only

## SHOW.DETAIL(rowcol,rowcol_num, expand)

Expands or collapses the detail under the specified expand or collapse button.

***Rowcol*** is a number that specifies whether to operate on rows or columns of data.

| Rowcol | Operates on |
|---|---|
| 1 | Rows |
| 2 | Columns |

***Rowcol_num*** is a number that specifies the row or column to expand or collapse. If you are in A1 mode, you must still give the column as a number. If *rowcol_num* is not a summary row or column, SHOW.DETAIL returns the #VALUE! error value and interrupts the macro.

***Expand*** is a logical value that specifies whether to expand or collapse the detail under the row or column. If *expand* is TRUE, Microsoft Excel expands the detail under the row or column; if FALSE, it collapses the detail under the row or column. If *expand* is omitted, the detail is expanded if it is currently collapsed and collapsed if it is currently expanded.

#### Examples

The following macro formula collapses the detail under column 4:

```
SHOW.DETAIL(2,4,FALSE)
```

The following macro formula either expands or collapses the detail under row 27, depending on its current state:

```
SHOW.DETAIL(1,27)
```

#### Related Function

| | |
|---|---|
| SHOW.LEVELS | Displays a specific number of levels of an outline |

Macro Sheets Only

## SHOW.INFO(logical)

Equivalent to the Window Show Info command. Controls the display of the Info window. For detailed information on the Window Show Info command, see online Help.

***Logical*** controls the display of the Info window.

- If *logical* is TRUE, Microsoft Excel activates the Info window.
- If the current window is the Info window and *logical* is FALSE, Microsoft Excel activates the document linked to the Info window.

#### Tip

Before using SHOW.INFO, use ACTIVATE to activate the document you want information about and SELECT or FORMULA.GOTO to select the cell or range you want information about.

#### Note

The Info window is not available when a chart is active. If SHOW.INFO is carried out when a chart is the active document, an Info window will be displayed for the worksheet or macro sheet that was most recently active.

#### Related Function

| | |
|---|---|
| GET.CELL | Returns information about the formatting, location, or contents of a cell |

Macro Sheets Only

## SHOW.LEVELS(row_level,col_level)

Displays the specified number of row and column levels of an outline.

***Row_level*** specifies the number of row levels of an outline to display. If the outline has fewer levels than specified by *row_level*, Microsoft Excel shows all levels. If *row_level* is omitted, no action is taken on rows.

***Col_level*** specifies the number of column levels of an outline to display. If the outline has fewer levels than specified by *col_level*, Microsoft Excel shows all levels. If *col_level* is omitted, no action is taken on columns.

#### Note

If you omit both arguments, SHOW.LEVELS returns the #VALUE! error value.

#### Example

The following macro formula displays row levels 1 through 3 and column level 1 of the outline:

```
SHOW.LEVELS(3,1)
```



#### Related Function

| | |
|---|---|
| SHOW.DETAIL | Expands or collapses a portion of an outline |

## SIGN(number)

Determines the sign of a number. Returns 1 if *number* is positive, 0 if *number* is 0, and –1 if *number* is negative.

#### Examples

SIGN(10) equals 1

SIGN(4–4) equals 0

SIGN(–0.00001) equals –1

#### Related Function

| | |
|---|---|
| ABS | Returns the absolute value of a number |

## SIN(number)

Returns the sine of the given angle.

***Number*** is the angle in radians for which you want the sine. If your argument is in degrees, multiply it by PI()/180 to convert it to radians.

#### Examples

SIN(PI()) equals 1.22E–16, which is approximately zero. The sine of π is zero.

SIN(PI()/2) equals 1

SIN(30*PI()/180) equals 0.5, the sine of 30 degrees

#### Related Functions

| | |
|---|---|
| ASIN | Returns the arcsine of a number |
| PI | Returns the value π |

## SINH(number)

Returns the hyperbolic sine of *number*.

The formula for the hyperbolic sine is:

$$\mathrm{SINH}(z) = \frac{e^{z} - e^{-z}}{2}$$

#### Examples

SINH(1) equals 1.175201194

SINH(–1) equals –1.175201194

You can use the hyperbolic sine function to approximate a cumulative probability distribution. Suppose a laboratory test value varies between 0 and 10 seconds. An empirical analysis of the collected history of experiments shows that the probability of obtaining a result, *x*, of less than *t* seconds is approximated by the following equation:

$P(x<t) = 2.868 * \mathrm{SINH}(0.0342 * t)$, where $0<t<10$

To calculate the probability of obtaining a result of less than 1.03 seconds, substitute 1.03 for *t*:

2.868*SINH(0.0342*1.03) equals 0.101049063

You can expect this result to occur about 101 times for every 1000 experiments.

#### Related Functions

| | |
|---|---|
| ASINH | Returns the inverse hyperbolic sine of a number |
| COSH | Returns the hyperbolic cosine of a number |
| TANH | Returns the hyperbolic tangent of a number |

Macro Sheets Only

## SIZE(width,height,window_text)

In Microsoft Excel for Windows and OS/2, equivalent to the Size command on the Control menu or to changing the size of a window by dragging its size box. In Microsoft Excel for the Macintosh, equivalent to changing the size of a window by dragging

its size box. SIZE changes the size of a window by moving its lower-right corner so that the window has the width and height you specify. SIZE does not change the position of the upper-left corner of the window or affect whether the specified window is active or inactive.

***Width*** specifies the width of the window and is measured in points.

***Height*** specifies the height of the window and is measured in points.

***Window_text*** specifies which window to size.

- *Window_text* is text enclosed in quotation marks or a reference to a cell containing text.
- If *window_text* is omitted, it is assumed to be the name of the active window.

### Notes

- In Microsoft Excel for Windows or OS/2, an error occurs if you try to resize a window that has already been enlarged to its maximum size. You must first restore the window to its original size using the FULL(FALSE) formula. For more information, see FULL.
- Most macro functions that describe movement or position on the screen take arguments measured in points. One point is equal to 1/72nd of an inch. Points are commonly used in typesetting—the height of fonts in Microsoft Excel, for example, is measured in points.

### Example

The following macro formula changes the size of the active window to 250 × 125 points:

```
SIZE(250,125)
```

### Related Functions

| | |
|---|---|
| FORMAT.SIZE | Changes the size of the selected object |
| MOVE | Moves a window |

## SLN(cost,salvage,life)

Returns the straight-line depreciation for an asset for a single period.

***Cost*** is the initial cost of the asset.

***Salvage*** is the value at the end of the depreciation (sometimes called the salvage value of the asset).

***Life*** is the number of periods over which the asset is being depreciated (sometimes called the useful life of the asset).

### Example

Suppose you've bought a truck for $30,000 that has a useful life of 10 years and a salvage value of $7,500. The depreciation allowance for each year is:

SLN(30000,7500,10) equals $2,250

### Related Functions

| | |
|---|---|
| DDB | Returns the depreciation for an asset for a specified period, using the double-declining balance method |
| SYD | Returns the sum-of-years' depreciation of an asset for a specified period |
| VDB | Returns the depreciation of an asset for a partial period |

*Macro Sheets Only*

## SORT(sort_by,key1,order1,key2,order2,key3,order3)
## SORT?(sort_by,key1,order1,key2,order2,key3,order3)

Equivalent to the Data Sort command. Sorts the rows or columns of the selection according to the contents of a key row or column within the selection. Use SORT to rearrange information into ascending or descending order.

***Sort_by*** is a number specifying whether to sort by rows or columns. Enter 1 to sort by rows or 2 to sort by columns.



***Key1*** is a reference to the cell or cells you want to use as the first sort key. The sort key identifies which column to sort by when sorting rows or which row to sort by when sorting columns. There are two ways to specify sort keys:

| Type of key | Examples |
|---|---|
| An external reference to the active worksheet. | !B:B or !Price |
| An R1C1-style reference in the form of text. If the reference is relative, it is assumed to be relative to the active cell in the selection. | "C2" or "C[1]" or "Price" |

***Order1*** specifies whether to sort the row or column containing *key1* in ascending or descending order. Enter 1 to sort in ascending order or 2 to sort in descending order.

***Key2***, ***order2***, ***key3***, and ***order3*** are similar to *key1* and *order1*. *Key2* specifies the second sort key, and *order2* specifies whether to sort the row or column containing *key2* in ascending or descending order. *Key3* and *order3* work similarly.

### Tip

If you want to sort using more than three keys, then sort the data three keys at a time, starting with the least important group of keys and progressing to the most important group, but listing the most important key first within each group. See the second example below.

### Examples

To sort the selected rows using columns A, B, and C as the sort keys so that the cells in columns A and B appear in ascending order and the cells in column C appear in descending order:

```
SORT(1,!$A$4,1,!$B$4,1,!$C$4,2)
```

The following lines in a macro sort the selected rows in ascending order using columns A, B, C, D, and E as the sort keys:

| | A | B | C |
|---|---|---|---|
| 13 | | =SORT(1,!$D$4,1,!$E$4,1) | |
| 14 | | =SORT(1,!$A$4,1,!$B$4,1,!$C$4,1) | |

*Macro Sheets Only*

## SPLIT(col_split,row_split)

In Microsoft Excel for Windows or OS/2, equivalent to the Split command on the Control menu for the document window or to dragging the split bar in the active window's scroll bar. In Microsoft Excel for the Macintosh, equivalent to dragging the split bar in the active window's scroll bar. SPLIT splits the active window into panes. Use SPLIT when you want to view different parts of the active document at the same time.

***Col_split*** specifies where to split the window vertically and is measured in columns from the left of the window.

***Row_split*** specifies where to split the window horizontally and is measured in rows from the top of the window.

If an argument is 0 and there is a split in that direction, the split is removed. If an argument is omitted, a split in that direction is not changed.

### Examples

The following macro formula splits the window horizontally after the third row and removes a vertical split, if there is one:

```
SPLIT(0,3)
```

The following macro formula splits the window vertically after the third column and has no effect on horizontal splitting:

```
SPLIT(3,)
```

### Related Function

| | |
|---|---|
| FREEZE.PANES | Freezes the panes of a window so that they do not scroll |

## SQRT(number)

Returns a positive square root.

***Number*** is the number for which you want the square root. If *number* is negative, SQRT returns the #NUM! error value.

### Examples

SQRT(16) equals 4

SQRT(–16) equals #NUM!

SQRT(ABS(–16)) equals 4

*Macro Sheets Only*

## STANDARD.FONT(font_text,size_num,bold, italic,underline,strike,color,outline,shadow)

Sets the attributes of the standard font (or Normal style) for the active worksheet or macro sheet. STANDARD.FONT is included for compatibility with earlier versions of Microsoft Excel. Normally, to define and apply a standard font style in Microsoft Excel version 3.0, you use the DEFINE.STYLE and APPLY.STYLE functions, which are equivalent to the actions you perform when you change the Normal style definition with the Format Styles command.

STANDARD.FONT takes the same arguments as FORMAT.FONT. For more information about the arguments, see FORMAT.FONT earlier in this manual. For more information on defining and applying styles, see DEFINE.STYLE and APPLY.STYLE earlier in this manual. Also see Chapter 5, "Formatting a Worksheet," in the *Microsoft Excel User's Guide*.

### Related Functions

| | |
|---|---|
| APPLY.STYLE | Applies a style to the selection |
| DEFINE.STYLE | Creates or changes a cell style |
| FORMAT.FONT | Applies a font to the current selection |

## STDEV(number1,number2,...)

Returns an estimate for the standard deviation of a population based on a sample given as arguments. The standard deviation is a measure of how widely values are dispersed from the average value (the mean).

***Number1,number2,...*** are 1 to 14 number arguments corresponding to a sample of a population. *Number* can be a reference to a range. All text, logical, or empty cell values cause errors.

### Notes

- STDEV assumes that its arguments are a sample of the population. If your data represents the entire population, you should compute the standard deviation using STDEVP.
- The standard deviation is calculated using the "nonbiased" or "*n*–1" method.
- STDEV uses the following formula:

$$\sqrt{\frac{n\sum x^2-(\sum x)^2}{n(n-1)}}$$

### Example

Suppose 10 tools stamped from the same machine during a production run are collected as a random sample and measured for breaking strength. The sample values (1345, 1301, 1368, 1322, 1310, 1370, 1318, 1350, 1303, 1299) are stored in cells A2:E3. STDEV estimates the standard deviation of breaking strengths for all the tools.

STDEV(A2:E3) equals 27.46

### Related Functions

| | |
|---|---|
| DSTDEV | Estimates the standard deviation from selected database entries |
| STDEVP | Calculates the true standard deviation when you have all the data for a population |
| VAR and VARP | Calculate variance |



## STDEVP(number1,number2,...)

Returns the standard deviation of a population given the entire population as arguments. The standard deviation is a measure of how widely values are dispersed from the average value (the mean).

***Number1,number2,...*** are 1 to 14 number arguments corresponding to a population. *Number* can be a reference to a range. All text, logical, or empty cell values cause errors.

### Notes

- STDEVP assumes that its arguments are the entire population. If your data represents a sample of the population, you should compute the standard deviation using STDEV.
- For large sample sizes, STDEV and STDEVP return approximately equal values.
- The standard deviation is calculated using the "biased" or "*n*" method.
- STDEVP uses the following formula:

$$\sqrt{\frac{n\sum x^2-(\sum x)^2}{n^2}}$$

### Example

Using the same data from the STDEV example and assuming that only 10 tools are produced during the production run, STDEVP measures the standard deviation of breaking strengths for all the tools.

STDEVP(A2:E3) equals 26.05

### Related Functions

| | |
|---|---|
| DSTDEV | Estimates the standard deviation from selected database entries |
| STDEV | Estimates the standard deviation for a population using a sample of the population |
| VAR and VARP | Calculate variance |

*Macro Sheets Only*

## STEP()

Stops the normal flow of a macro and calculates it one cell at a time. Running a macro one cell at a time is called single-stepping and is very useful when you are debugging a macro. Use the STEP function, instead of choosing the Macro Run command and selecting the Step check box, when you want to start single-stepping at a specific line in a macro. For more information on debugging macros, see Chapter 19, "Designing and Writing a Command Macro," in the *Microsoft Excel User's Guide* and the Macro Debugger macro in the Macro Library in online Help.

When Microsoft Excel encounters a STEP function, it stops running the macro and displays a dialog box. The dialog box tells you which cell in the macro Microsoft Excel is about to calculate, and what formula is in that cell. You can choose Step to carry out the next instruction; choose Evaluate to calculate part of the formula; choose Halt to interrupt the macro; or choose Continue to continue the macro without single-stepping. You can also evaluate the formula by holding down the SHIFT key while you click the Step button.

### Notes

- When placed at the beginning of a macro, STEP is equivalent to choosing the Macro Run command and selecting the Step check box in the Run dialog box.
- You can start single-stepping while a macro is running by pressing ESC in Microsoft Excel for Windows or OS/2 or by pressing ESC or COMMAND+PERIOD in Microsoft Excel for the Macintosh.
- The Single Step dialog box is initially displayed in the lower-right corner of the screen. You can move the dialog box if it's in your way. If you move it, it will remain in the new location until you stop single-stepping.

### Tip

When the Single Step dialog box is displayed, if you hold down the SHIFT key when you choose Halt, Microsoft Excel halts the macro and goes to the last cell calculated.

***Related Functions***

| | |
|---|---|
| HALT | Halts a macro |
| RUN | Runs a macro |

*Macro Sheets Only*

## *STYLE(bold,italic)*
## *STYLE?(bold,italic)*

This macro function is included for compatibility with macros written with Microsoft Excel for the Macintosh, version 1.5 or earlier. If you want to change a font to bold or italic, use the FORMAT.FONT function. For more information, see FORMAT.FONT earlier in this manual.

STYLE checks the fonts for a bold and/or italic font and applies it to the current selection. If no appropriate font is available, Microsoft Excel finds the most similar font available and formats the selection using that font.

***Bold*** is a logical value specifying whether to apply a bold font. If *bold* is TRUE, Microsoft Excel formats the current selection using an available bold font; if FALSE, Microsoft Excel applies a nonbold font.

***Italic*** is a logical value specifying whether to apply an italic font. If *italic* is TRUE, Microsoft Excel formats the current selection using an available italic font; if FALSE, Microsoft Excel applies a nonitalic font.

*Macro Sheets Only*

## *SUBSCRIBE.TO(file_text,format_num)*

Inserts the contents of the edition into the active document at the current selection. Use SUBSCRIBE.TO to incorporate editions published from other documents into your Microsoft Excel worksheets and macro sheets. SUBSCRIBE.TO returns TRUE if successful.

***Note***

This function is only available if you are using Microsoft Excel for the Macintosh with system software version 7.0 or later.

***File_text*** is the name, as a quoted text string, of the edition you want to insert into the active document. Unless *file_text* is in the current folder, supply the full pathname of the document. If *file_text* cannot be found, SUBSCRIBE.TO returns the #VALUE! error value and interrupts the macro.

***Notes***

- If a single cell is selected, the data from the edition file is placed into as large a range of cells as is required by the data. Data already present in those cells is replaced. If the data is a picture, it is inserted from the upper-left corner of the selected cell.
- If a range of cells is selected, and the range is not big enough to contain the edition data, SUBSCRIBE.TO returns the #VALUE! error value.

***Format_num*** is the number 1 or 2 and specifies the format type of the file you are subscribing to.

| *Format_num* | Format type |
|---|---|
| 1 or omitted | Picture |
| 2 | Text (includes BIFF, VALU, TEXT, and CSV formats) |

***Examples***

The following macro formula subscribes to an edition named Corporate Logo in the current folder. The 1 indicates that the edition is a picture, if available.

```
SUBSCRIBE.TO("Corporate Logo",1)
```

The following macro formula subscribes to an edition in the specified path named Sales Heading. The 2 indicates that the edition is text, if available.

```
SUBSCRIBE.TO("Hard Disk:Word:Sales Heading",2)
```



***Related Functions***

| | |
|---|---|
| CREATE.PUBLISHER | Creates a publisher in a Microsoft Excel document |
| EDITION.OPTIONS | Sets publisher and subscriber options |
| GET.LINK.INFO | Returns information about a link |

## *SUBSTITUTE(text,old_text,new_text, instance_num)*

Substitutes *new_text* for *old_text* in a text string. Use SUBSTITUTE when you want to replace specific text in a text string; use REPLACE when you want to replace any text that occurs in a specific location in a text string.

***Text*** is the text enclosed in quotation marks or the reference to a cell containing text for which you want to substitute characters.

***Old_text*** is the text you want to replace.

***New_text*** is the text you want to replace *old_text* with.

***Instance_num*** specifies which occurrence of *old_text* you want to replace with *new_text*. If you specify *instance_num*, only that instance of *old_text* is replaced. Otherwise, every occurrence of *old_text* in *text* is changed to *new_text*.

***Examples***

```
SUBSTITUTE("Sales Data","Sales","Cost") equals "Cost Data"

SUBSTITUTE("Quarter 1, 1991","1","2",1) equals "Quarter 2, 1991"

SUBSTITUTE("Quarter 1, 1991","1","2",3) equals "Quarter 1, 1992"
```

To replace every occurrence of the text constant named Separator in the cell named CellCont2 with square brackets (this formula appears in the PARSE macro example):

```
SUBSTITUTE(CellCont2,Separator,"] [")
```

***Related Functions***

| | |
|---|---|
| REPLACE | Replaces characters in a string of text |
| TRIM | Removes spaces from text |

## *SUM(number1,number2,...)*

Returns the sum of all the numbers in the list of arguments.

***Number1,number2,...*** are 1 to 14 arguments for which you want the sum.

- Numbers, logical values, and text representations of numbers that you type directly into the list of arguments are counted. See the first and second examples below.
- If an argument is an array or reference, only numbers in that array or reference are counted. Empty cells, logical values, text, or error values in the array or reference are ignored. See the third example below.
- Arguments that are error values or text that cannot be translated into numbers cause errors.

***Note***

You can also enter the SUM function with the autosum button on the tool bar. For more information, see Chapter 3, "Creating and Using a Worksheet," in the *Microsoft Excel User's Guide*.

***Examples***

```
SUM(3,2) equals 5

SUM("3",2,TRUE) equals 6 because the text values are translated into numbers, and the logical value TRUE is translated into the number 1.
```

Unlike the previous example, if A1 contains "3" and B1 contains TRUE, then:

```
SUM(A1,B1,2) equals 2 because references to non-number values in references are not translated.
```

If cells A2:E2 contain 5, 15, 30, 40, and 50:
SUM(A2:C2) equals 50
SUM(B2:E2,15) equals 150

### Related Functions

| | |
|---|---|
| AVERAGE | Returns the average of its arguments |
| COUNT and COUNTA | Count how many numbers or arguments are in a list of arguments |
| PRODUCT | Multiplies its arguments |
| SUMPRODUCT | Returns the sum of the products of corresponding array components |

## SUMPRODUCT(array1,array2,...)

Multiplies corresponding components in the given arrays and returns the sum of those products.

***Array1, array2,...*** are 2 to 14 arrays whose components you want to multiply and then add.

- The array arguments must have the same dimensions. If they do not, SUMPRODUCT returns the #VALUE! error value.
- SUMPRODUCT treats array entries that are not numeric as if they were zeros.

### Example

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | 3 | 4 | | 2 | 7 | |
| 2 | 8 | 6 | | 6 | 7 | |
| 3 | 1 | 9 | | 5 | 3 | |
| 4 | | | | | | |

The following formula multiplies all the components of the two arrays in the preceding worksheet and then adds the products—that is, 3*2 + 4*7 + 8*6 + 6*7 + 1*5 + 9*3.

SUMPRODUCT(A1:B3,D1:E3) equals 156

### Note

The preceding example returns the same result as the formula SUM(A1:B3*D1:E3) entered as an array. Using arrays provides a more general solution for doing operations similar to SUMPRODUCT. For example, you can calculate the sum of the squares of the elements in A1:B3 by using the formula SUM(A1:B3^2) entered as an array.

### Related Functions

| | |
|---|---|
| MMULT | Returns the matrix product of two arrays |
| PRODUCT | Multiplies its arguments |
| SUM | Adds its arguments |

## SYD(cost,salvage,life,per)

Returns the sum-of-years' digits depreciation for an asset for a specified period.

***Cost*** is the initial cost of the asset.

***Salvage*** is the value at the end of the depreciation (sometimes called the salvage value of the asset).

***Life*** is the number of periods over which the asset is being depreciated (sometimes called the useful life of the asset).

***Per*** is the period and must use the same units as *life*.

### Examples

If you've bought a truck for $30,000 that has a useful life of 10 years and a salvage value of $7,500, the yearly depreciation allowance for the first year is:

SYD(30000,7500,10,1) equals $4,090.91

The yearly depreciation allowance for the 10th year is:

SYD(30000,7500,10,10) equals $409.09



### Related Functions

| | |
|---|---|
| DDB | Returns the depreciation of an asset for a specified period, using the double-declining balance method |
| SLN | Returns the straight-line depreciation of an asset for one period |
| VDB | Returns the depreciation of an asset for a partial period |

## T(value)

Returns the text referred to by *value*.

***Value*** is the value you want to test. If *value* is or refers to text, T returns *value*. If *value* does not refer to text, T returns "" (empty text).

### Note

You do not generally need to use the T function in a formula since Microsoft Excel automatically converts values as necessary. This function is provided for compatibility with other spreadsheet programs.

### Examples

If B1 contains the text "Rainfall":
T(B1) equals "Rainfall"

If B2 contains the number 19:
T(B2) equals ""

T("TRUE") equals "TRUE"

T(TRUE) equals ""

### Related Functions

| | |
|---|---|
| CELL | Returns information about the formatting, location, or contents of a cell |
| N | Converts its argument to a number |
| VALUE | Converts a text argument to a number |

*Macro Sheets Only*

## TABLE(row_ref,column_ref)
## TABLE?(row_ref,column_ref)

Equivalent to the Data Table command (full menus). Creates a table based on the input values and formulas you define on a worksheet. Use data tables to perform a "what-if" analysis by changing certain constant values on your worksheet to see how values in other cells are affected.

For information on data tables, see Chapter 7, "Analyzing and Calculating a Worksheet," in the *Microsoft Excel User's Guide*.

***Row_ref*** specifies the one cell to use as the row input for your table.

- *Row_ref* should be either an external reference to a single cell on the active worksheet, such as !$A$1 or !Price, or an R1C1-style reference to a single cell in the form of text, such as "R1C1", "R[–1]C[–1]", or "Price".
- If *row_ref* is an R1C1-style reference, it is assumed to be relative to the active cell in the selection.

***Column_ref*** specifies the one cell to use as the column input for your table. *Column_ref* is subject to the same restrictions as *row_ref*, listed above.

### Examples

Each of the following macro formulas creates a two-input table in the selection on the active sheet using A1 and A2 as input cells:

TABLE(!$A$1,!$A$2)
TABLE("R1C1","R2C1")

If the selection is C3:E5 and the active cell is C3, this macro formula creates a one-input table in the selection using cell B2 as the column-input cell:

TABLE(,"R[–1]C[–1]")

If cell B2 on the active worksheet is named Size, each of the following macro formulas creates a one-input table in the selection using cell B2 as the row-input cell:

TABLE(!Size,)
TABLE("Size",)

If cell B2 on the active worksheet contains the text "Size," the first macro formula below sets up a one-input table in the selection using cell B2 as the row-input cell, while the second macro formula sets up a one-input table using a cell named Size as the row-input cell.

TABLE(!$B$2,)

TABLE(DEREF(!$B$2),)

## TAN(number)

Returns the tangent of the given angle.

***Number*** is the angle in radians for which you want the tangent. If your argument is in degrees, multiply it by PI()/180 to convert it to radians.

### Examples

TAN(0.785) equals 0.9920399

TAN(45*PI()/180) equals 1

### Related Functions

| | |
|---|---|
| ATAN | Returns the arctangent of a number |
| ATAN2 | Returns the arctangent from x- and y-coordinates |
| PI | Returns the value π |

## TANH(number)

Returns the hyperbolic tangent of *number*.

The formula for the hyperbolic tangent is:

$$TANH(z) = \frac{SINH(z)}{COSH(z)}$$

### Examples

TANH(–2) equals –0.96402758

TANH(0) equals 0

TANH(0.5) equals 0.462117157

### Related Functions

| | |
|---|---|
| ATANH | Returns the inverse hyperbolic tangent of a number |
| COSH | Returns the hyperbolic cosine of a number |
| SINH | Returns the hyperbolic sine of a number |

Macro Sheets Only

## TERMINATE(channel_num)

Closes a DDE channel previously opened with the INITIATE function. Use TERMINATE to close a channel after you have finished communicating with another application.

### Note

Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

***Channel_num*** is the number returned by a previously run INITIATE function. *Channel_num* identifies a DDE channel to close.

If TERMINATE is not successful, it returns the #VALUE! error value.

For information on accessing other applications, see Chapter 8, "Working with Data from Multiple Documents," in the *Microsoft Excel User's Guide*.

### Related Functions

| | |
|---|---|
| EXEC | Starts another application |
| INITIATE | Opens a channel to another application |

## TEXT(value,format_text)

Converts a value to text in a specific number format.



***Value*** is a numeric value, a formula that evaluates to a numeric value, or a reference to a cell containing a numeric value.

***Format_text*** is a quoted number format from the Format Number dialog box. *Format_text* cannot contain an asterisk (*) and cannot be "General." For information on the Format Number command and a list of the available number formats, see Chapter 5, "Formatting a Worksheet," in the *Microsoft Excel User's Guide*.

### Note

Formatting a cell with the Format Number command changes only the format, not the value. Using the TEXT function converts a value to formatted text, and the result is no longer calculated as a number.

### Examples

TEXT(2.715,"$0.00") equals "$2.72"

TEXT("4/15/91","mmmm dd, yyyy") equals "April 15, 1991"

### Related Functions

| | |
|---|---|
| DOLLAR | Formats a number and converts it to text |
| FIXED | Formats a number as text with a fixed number of decimals |
| FORMAT.NUMBER | Formats numbers, dates, and times |
| T | Converts its arguments to text |
| VALUE | Converts a text argument to a number |

Macro Sheets Only

## TEXT.BOX(add_text,object_id_text,start_num,num_chars)

Replaces characters in a text box with the text you specify. Use TEXT.BOX to update text boxes with new text.

***Add_text*** is the text you want to add to the text box.

***Object_id_text*** is the name of the text box to which you want to add text (for example, "Text 1" or "Text 2"). If *object_id_text* is omitted, it is assumed to be the selected text box.

***Start_num*** is a number specifying the position of the first character you want to replace (or the position at which you want to insert characters if you do not want to replace any). If *start_num* is omitted, it is assumed to be 1.

***Num_chars*** is the number of characters you want to replace. If *num_chars* is 0, then no characters are replaced, and *add_text* is inserted starting at the position *start_num*. If *num_chars* is omitted, all the characters are replaced.

### Examples

The following macro formula replaces the first five characters in a text box named "Text 5" with the text "Net Income":

TEXT.BOX("Net Income","Text 5",1,5)

The following macro formula inserts the words "Account Summary for 1991" at the beginning of a text box named "Text 6":

TEXT.BOX("Account Summary for 1991","Text 6",1,0)

### Related Functions

| | |
|---|---|
| CREATE.OBJECT | Creates the specified object |
| FORMAT.FONT | Applies a font to the current selection |

Macro Sheets Only

## TEXTREF(text,a1)

Converts text to an absolute A1- or R1C1-style reference. Use TEXTREF to convert references stored as text to references so that you can use them with other functions, such as SELECT.

***Text*** is a reference in the form of text.

*A1* is a logical value specifying the reference type of *text*. If *a1* is TRUE, *text* is assumed to be an A1-style reference; if FALSE or omitted, *text* is assumed to be an R1C1-style reference.

*Notes*

- If you use TEXTREF, you will usually get the value contained in the cell specified by *text*, not the reference itself, because references are automatically converted into the contents of the referenced cell.
- If you use TEXTREF as a *reference* argument to a function, then Microsoft Excel does not convert the reference to a value.

*Tip*

You can convert a reference to text with REFTEXT, manipulate it with the REPLACE and MID functions, and convert it back to a reference with TEXTREF.

***Examples***

TEXTREF("B7",TRUE) equals the reference value $B$7

TEXTREF("R5C5",FALSE) equals the reference value R5C5

TEXTREF("B7",FALSE) equals the #REF! error value, because "B7" can't be interpreted as an R1C1-style reference.

The following macro uses REFTEXT to convert the reference returned by SELECTION into text. The SUBSTITUTE function then changes the worksheet name in the reference, and the TEXTREF function in cell B4 converts the text returned by SUBSTITUTE back to a reference, to be used by FORMULA.FILL.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | ConvertAndFill | |
| 4 | OldRef | =REFTEXT(SELECTION()) | Gets reference of selection |
| 5 | NewRef | =SUBSTITUTE(OldRef,"Sheet1","Sheet2") | Changes document names |
| 6 | | =FORMULA.FILL("text",TEXTREF(NewRef,)) | Fills into new reference |
| 7 | | =RETURN() | |

Also see the example in REFTEXT.

***Related Functions***

| | |
|---|---|
| DEREF | Returns the values of the cells in a reference |
| INDIRECT | Returns a reference indicated by a text value |
| OFFSET | Returns a reference offset from a given reference |
| REFTEXT | Converts a reference into text |

## TIME(hour,minute,second)

Returns the serial number of a particular time. The serial number returned by TIME is a decimal fraction ranging from 0 to 0.99999999, representing the times from 0:00:00 (12:00:00 A.M.) to 23:59:59 (11:59:59 P.M.).

***Hour*** is a number from 0 to 23 representing the hour.

***Minute*** is a number from 0 to 59 representing the minute.

***Second*** is a number from 0 to 59 representing the second.

***Note***

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information on date systems and serial numbers, see NOW.

***Examples***

TIME(12,0,0) equals the serial number 0.5, which is equivalent to 12:00:00 P.M.

TIME(16,48,10) equals the serial number 0.700115741, which is equivalent to 4:48:10 P.M.

TEXT(TIME(23,18,14),"h:mm:ss AM/PM") equals "11:18:14 PM"



***Related Functions***

| | |
|---|---|
| HOUR, MINUTE, and SECOND | Convert serial numbers into hours, minutes, and seconds |
| NOW | Returns the serial number of the current date and time |

## TIMEVALUE(time_text)

Returns the serial number of the time represented by *time_text*. The serial number is a decimal fraction ranging from 0 to 0.99999999, representing the times from 0:00:00 (12:00:00 A.M.) to 23:59:59 (11:59:59 P.M.). Use TIMEVALUE to convert a time represented as text into a serial number.

***Time_text*** is a quoted text string that gives a time in any one of Microsoft Excel's time formats. Date information in *time_text* is ignored.

***Note***

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information on date systems and serial numbers, see NOW.

***Examples***

TIMEVALUE("2:24 AM") equals 0.1

TIMEVALUE("22-Aug-55 6:35 AM") equals 0.274305556

***Related Functions***

| | |
|---|---|
| DATEVALUE | Converts a date in the form of text to a serial number |
| HOUR, MINUTE, and SECOND | Convert serial numbers into hours, minutes, and seconds |
| NOW | Returns the serial number of the current date and time |
| TIME | Returns the serial number of a particular time |

## TODAY()

Returns the serial number of the current date. The serial number is the date-time code used by Microsoft Excel for date and time calculations. For more information about serial numbers, see NOW.

***Related Functions***

| | |
|---|---|
| DATE | Returns the serial number of a specified date |
| DAY | Converts a serial number into a day |
| NOW | Returns the serial number of the current date and time |

## TRANSPOSE(array)

Returns the transpose of *array*. TRANSPOSE must be entered as an array formula in a range that has the same number of rows and columns, respectively, as *array* has columns and rows. Use TRANSPOSE to shift the vertical and horizontal orientation of an array on a worksheet or macro sheet. For example, some functions, such as DOCUMENTS, return horizontal arrays. The following formula would return a vertical array from DOCUMENTS:

TRANSPOSE(DOCUMENTS())

***Array*** is an array on a worksheet or macro sheet that you want to transpose.

The transpose of an array is created by using the first row of the array as the first column of the new array, the second row of the array as the second column of the new array, and so on.

*Tip*

To transpose an array and paste only the values contained in the array, use the PASTE.SPECIAL macro function with 3 (values) as the *paste_num* argument and TRUE as the *transpose* argument.

### Example

| | A | B | C | D |
|---|---|---|---|---|
| 1 | 1 | 2 | 3 | |
| 2 | 4 | 5 | 6 | |
| 3 | | | | |
| 4 | 1 | 4 | | |
| 5 | 2 | 5 | | |
| 6 | 3 | 6 | | |

In the worksheet above, when the following formula is entered as an array:

TRANSPOSE($A$1:$C$2) equals the values in A4:B6

### Related Functions

| | |
|---|---|
| MDETERM | Returns the matrix determinant of an array |
| MINVERSE | Returns the matrix inverse of an array |
| MMULT | Returns the matrix product of two arrays |

## TREND(known_y's,known_x's,new_x's,const)

Fits a straight line (using the method of least squares) to the arrays *known_y's* and *known_x's*. Then it returns the *y*-values along that line for the array of *new_x's* that you specify.

***Known_y's*** is the set of *y*-values you already know in the relationship $y = mx + b$.

- If the array *known_y's* is in a single column, then each column of *known_x's* is interpreted as a separate variable.
- If the array *known_y's* is in a single row, then each row of *known_x's* is interpreted as a separate variable.

***Known_x's*** is an optional set of *x*-values that you may already know in the relationship $y = mx + b$.

- The array *known_x's* can include one or more sets of variables. If only one variable is used, *known_y's* and *known_x's* can be ranges of any shape, as long as they have equal dimensions. If more than one variable is used, *known_y's* must be a vector (that is, a range with a height of one row or a width of one column).
- If *known_x's* is omitted, it is assumed to be the array {1,2,3,...} that is the same size as *known_y's*.

***New_x's*** are new *x*-values for which you want TREND to return corresponding *y*-values.

- *New_x's* must include a column (or row) for each independent variable, just as *known_x's* does. So, if *known_y's* is in a single column, *known_x's* and *new_x's* should have the same number of columns. If *known_y's* is in a single row, *known_x's* and *new_x's* should have the same number of rows.
- If you omit *new_x's*, it is assumed to be the same as *known_x's*.
- If you omit both *known_x's* and *new_x's*, they are assumed to be the array {1,2,3,...} that is the same size as *known_y's*.

***Const*** is a logical value specifying whether to force the constant *b* to equal 0.

- If *const* is TRUE or omitted, *b* is calculated normally.
- If *const* is FALSE, *b* is set equal to 0 and the *m*-values are adjusted so that $y = mx$.

### Notes

- For information on how Microsoft Excel fits a line to data, see LINEST earlier in this manual.
- You can use TREND for polynomial curve fitting by regressing against the same variable raised to different powers. For example, suppose column A contained *y*-values and column B contained *x*-values. You could enter *x*^2 in column C, *x*^3 in column D, and so on, and then regress columns B through D against column A.
- Formulas that return arrays must be entered as array formulas.
- When entering an array constant for an argument such as *known_x's*, use commas to separate values in the same row and semicolons to separate rows.



### Examples

Suppose a business wants to purchase a tract of land in July, the start of the next fiscal year. The business collected cost information that covers the most recent 12 months for a typical tract in the desired area.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | Month | Price | | |
| 2 | March | $133,890 | | |
| 3 | April | $135,000 | | |
| 4 | May | $135,790 | | |
| 5 | June | $137,300 | | |
| 6 | July | $138,130 | | |
| 7 | August | $139,100 | | |
| 8 | September | $139,900 | | |
| 9 | October | $141,120 | | |
| 10 | November | $141,890 | | |
| 11 | December | $143,230 | | |
| 12 | January | $144,000 | | |
| 13 | February | $145,290 | | |

When entered as a vertical array in the range C2:C6, the following formula returns the predicted prices for March, April, May, June, and July:

TREND(B2:B13,,{13;14;15;16;17}) equals {146172;147190;148208;149226;150244}

The company can expect a typical tract of land to cost about $150,244 if it waits until July. The preceding formula uses the default array {1;2;3;4;5;6;7;8;9;10;11;12} for the *known_x's* argument, corresponding to the 12 months of sales data. The array {13;14;15;16;17} corresponds to the next five months.

### Related Functions

| | |
|---|---|
| GROWTH and LOGEST | Are similar to TREND and LINEST, but fit your data to an exponential curve |
| LINEST | Also calculates a line, but returns the parameters of the line instead of an array of *y*-values |

## TRIM(text)

Removes all spaces from *text* except for single spaces between words. Use TRIM on text that you have received from another application that may have irregular spacing.

***Text*** is the text from which you want spaces removed. For information on using text arguments, see "Converting Data Types" earlier in this book.

### Example

TRIM(" First Quarter Earnings ") equals "First Quarter Earnings"

### Related Functions

| | |
|---|---|
| CLEAN | Removes control characters from text |
| MID | Extracts a number of characters from text |
| REPLACE and SUBSTITUTE | Replace characters within text |

## TRUE()

Returns the logical value TRUE.

### Note

You can enter the value TRUE directly into cells and formulas without using this function. The TRUE function is provided primarily for compatibility with other spreadsheet programs.

## TRUNC(number, num_digits)

Truncates a number to an integer by removing the fractional part of the number.

***Number*** is the number you want to truncate. For information on using number arguments, see "Converting Data Types" earlier in this book.

***Num_digits*** is a number specifying the precision of the truncation. The default value for *num_digits* is zero.

### Note

TRUNC and INT are similar in that both return integers. TRUNC removes the fractional part of the number. INT rounds numbers down to the nearest integer based on the value of the fractional part of the number. INT and TRUNC are different only when using negative numbers: TRUNC(–4.3) returns –4, but INT(–4.3) returns –5, because –5 is the lower number.

### Examples

TRUNC(8.9) equals 8

TRUNC(–8.9) equals –8

TRUNC(PI()) equals 3

### Related Functions

| | |
|---|---|
| INT | Rounds a number down to the nearest integer |
| MOD | Gives the remainder from division |
| ROUND | Rounds a number to a specified number of digits |

## TYPE(*value*)

Returns the type of *value*. Use TYPE when the behavior of another function depends on the type of value in a particular cell.

***Value*** can be any Microsoft Excel value, such as a number, text, logical value, and so on.

| If *value* is | TYPE returns |
|---|---|
| A number | 1 |
| Text | 2 |
| A logical value | 4 |
| A formula | 8 |
| An error value | 16 |
| An array | 64 |

### Note

TYPE is most useful when you are using functions that can accept a variety of different types of data, such as ARGUMENT and INPUT. Use TYPE to find out what type of data was returned by the function.

### Examples

If A1 contains the text Smith, then:

TYPE(A1) equals TYPE("Smith") equals 2

TYPE("MR. "&A1) equals 2

TYPE(2+A1) equals TYPE(#VALUE!) equals 16

TYPE({1,2;3,4}) equals 64

The following macro demonstrates how you can use TYPE to return the type of an argument given to a custom function.

| | | B | C |
|---|---|---|---|
| 1 | | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | Custom Function | |
| 4 | | =ARGUMENT("Value",15) | Name and type of argument |
| 5 | ArgType | =TYPE(Value) | Gets type of argument |
| 6 | | =IF(ArgType<>1,RETURN(""),RETURN(Value | Returns argument to function |
| 7 | | =RETURN() | |

The complete formula in cell B6 is:

```
=IF(ArgType<>1,RETURN(""),RETURN(Value*3))
```

### Related Functions

| | |
|---|---|
| GET.CELL | Returns information about the formatting, location, or contents of a cell |
| IS functions | Return information about the type of information in a cell |



*Macro Sheets Only*

## UNDO()

Equivalent to the Edit Undo command. Reverses certain actions and commands. UNDO is available in the same situations as the Edit Undo command. For detailed information on the Edit Undo command, see online Help.

*Macro Sheets Only*

## UNGROUP()

Equivalent to the Format Ungroup command. Separates a grouped object into individual objects. Use UNGROUP to separate a group of objects so that you can format, move, or size one of the objects. For detailed information on the Format Ungroup command, see online Help.

If the selection is not a grouped object, UNGROUP returns the #VALUE! error value and interrupts the macro.

### Related Function

| | |
|---|---|
| GROUP | Groups selected objects |

*Macro Sheets Only*

## UNHIDE(*window_text*)

Equivalent to the Window Unhide command (full menus). Use UNHIDE to display hidden windows. For detailed information on the Window Unhide command, see online Help.

***Window_text*** is the name of the window to unhide. If *window_text* is not the name of an open document, an error value is returned and the macro is interrupted. You cannot unhide a window of an add-in document.

### Examples

The following macro formula displays an embedded chart window:

UNHIDE("Sheet1 Chart2")

You can use GET.WINDOW(7,*window_text*) to find out if a specific window is hidden. The following macro displays the names of all currently hidden documents:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | UnhideAllWindows | Unhides all open windows |
| 4 | | =SET.NAME("WindowArray",WINDOWS()) | Names of open windows |
| 5 | | =FOR("Counter",1,COLUMNS(WindowArray)) | Begins a FOR–NEXT loop |
| 6 | WinName | =INDEX(WindowArray,1,Counter) | Returns one window name |
| 7 | Hidden | =GET.WINDOW(7,WinName) | Hidden sheet? |
| 8 | | =IF(Hidden,UNHIDE(WinName)) | Unhides hidden sheets |
| 9 | | =NEXT() | Continues the loop |
| 10 | | =RETURN() | |

### Related Functions

| | |
|---|---|
| GET.WINDOW | Returns information about a window |
| HIDE | Hides the active window |

*Macro Sheets Only*

## UNLOCKED.NEXT()
## UNLOCKED.PREV()

Equivalent to pressing TAB or SHIFT+TAB to move to the next or previous unlocked cell in a protected worksheet. Use these functions when you want to control which cell is active on a protected sheet.

### Related Functions

| | |
|---|---|
| CELL.PROTECTION | Controls protection and display of cells |
| PROTECT.DOCUMENT | Turns on and off cell and window protection |

Macro Sheets Only

## UNREGISTER(register_number)

Removes a dynamic link library or code resource that was previously registered in your computer's memory by the REGISTER macro function. Use UNREGISTER to free memory after you have finished using the code resources from a REGISTER function. For more information on using the CALL and REGISTER functions, see Appendix A, "Using the CALL and REGISTER Functions."

***Register_number*** is the value returned by the REGISTER function, which represents the code resource to be removed from memory.

### Related Functions

| | |
|---|---|
| CALL | Calls a procedure in a dynamic link library or code resource |
| REGISTER | Accesses a dynamic link library or code resource |

Macro Sheets Only

## UPDATE.LINK(link_text,type_of_link)

Equivalent to choosing the Update button with a link selected in the File Links dialog box. Updates a link to another document. Use UPDATE.LINK to get the newest information from a supporting document. For detailed information on the File Links command, see online Help.

***Link_text*** is a quoted text string describing the full pathname of the link as displayed in the File Links dialog box. If *link_text* is omitted, only links from the active document to other Microsoft Excel documents are updated.

***Type_of_link*** is a number from 1 to 4 that specifies the type of link to update.

| *Type_of_link* | Link document type |
|---|---|
| 1 or omitted | Microsoft Excel link |
| 2 | DDE link |
| 3 | Not available |
| 4 | Outgoing NewWave link |

### Examples

In Microsoft Excel for Windows or OS/2, the following macro formula updates the link from the document named SALES.XLS.

```
UPDATE.LINK("C:\EXCEL\SALES.XLS")
```

In Microsoft Excel for the Macintosh, the following macro formula updates the link from the document named Sales East.

```
UPDATE.LINK("Hard Disk:Excel:Sales East")
```

### Related Functions

| | |
|---|---|
| CHANGE.LINK | Changes a supporting worksheet link |
| GET.LINK.INFO | Returns information about a link |
| OPEN.LINKS | Opens a specified supporting document |

## UPPER(text)

Converts *text* to uppercase.

***Text*** is the text you want converted to uppercase. For information on text arguments, see "Converting Data Types" earlier in this book.

### Examples

The following macro formula converts the contents of the active cell to uppercase:

```
FORMULA(UPPER(ACTIVE.CELL()),ACTIVE.CELL())
```

The following macro uses UPPER along with the FIND and MID functions to ensure that the state abbreviations are capitalized.



| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **UppercaseState** | |
| 4 | Text | =INPUT("Enter city and state",2) | |
| 5 | Comma | =FIND(",",Text) | Gets location of the comma |
| 6 | Upper | =UPPER(MID(Text,Comma+1,2)) | Converts to uppercase |
| 7 | | =FORMULA(MID(Text,1,Comma)&Upper,Text) | Replaces "Text" with chars |
| 8 | | =RETURN() | |

Also see the example in PROPER.

### Related Functions

| | |
|---|---|
| LOWER | Converts text to lowercase |
| PROPER | Converts text to initial capitals |

## VALUE(text)

Converts *text* to a number.

***Text*** is the text enclosed in quotation marks or a reference to a cell containing the text you want to convert. *Text* can be in any of the constant number, date, or time formats recognized by Microsoft Excel. If *text* is not in one of these formats, VALUE returns the #VALUE! error value.

### Note

You do not generally need to use the VALUE function in a formula since Microsoft Excel automatically converts text to numbers as necessary. This function is provided for compatibility with other spreadsheet programs.

### Examples

```
VALUE("$1,000") equals 1,000
```

VALUE("16:48:00")–VALUE("12:00:00") equals "16:48:00"–"12:00:00" equals 0.2, the serial number equivalent to 4 hours and 48 minutes.

### Related Functions

| | |
|---|---|
| DOLLAR | Formats a number and converts it to text |
| FIXED | Rounds and formats a number and converts it to text |
| TEXT | Formats a number and converts it to text |

## VAR(number1,number2,...)

Returns an estimate for the variance of a population based on a sample given as arguments.

***Number1,number2,...*** are 1 to 14 number arguments corresponding to a sample of a population. All text, logical, or empty cell values cause errors.

### Notes

- VAR assumes that its arguments are a sample of the population. If your data represents the entire population, you should compute the variance using VARP.
- VAR uses the following formula:

$$\frac{n\sum x^2 - (\sum x)^2}{n(n-1)}$$

### Example

Suppose 10 tools stamped from the same machine during a production run are collected as a random sample and measured for breaking strength. The sample values (1345, 1301, 1368, 1322, 1310, 1370, 1318, 1350, 1303, 1299) are stored in cells A2:E3. VAR estimates the variance for the breaking strength of the tools.

```
VAR(A2:E3) equals 754.3
```

### Related Functions

| | |
|---|---|
| DVAR | Calculates variance from selected database entries |
| STDEV and STDEVP | Calculate standard deviation |
| VARP | Calculates the variance when you have all the data for a population |

## VARP(number1,number2,...)

Returns the variance of a population given the entire population as arguments.

**Number1,number2,...** are 1 to 14 number arguments corresponding to a population. All text, logical, or empty cell values cause errors.

### Note

VARP assumes that its arguments are the entire population. If your data represents a sample of the population, you should compute the variance using VAR.

### Example

Using the data from the VAR example and assuming that only 10 tools are produced during the production run, VARP measures the variance of breaking strengths for all the tools.

VARP(A2:E3) equals 678.8

### Related Functions

| | |
|---|---|
| DVARP | Calculates variance from selected database entries |
| STDEVP and STDEV | Calculate standard deviation |
| VAR | Estimates the variance from a sample |

## VDB(cost,salvage,life,start_period,end_period, factor,no_switch)

Returns the depreciation of an asset for any period you specify, including partial periods, using the double-declining balance method or some other method you specify. VDB stands for variable declining balance.

**Cost** is the initial cost of the asset.

**Salvage** is the value at the end of the depreciation (sometimes called the salvage value of the asset).

**Life** is the number of periods over which the asset is being depreciated (sometimes called the useful life of the asset).

**Start_period** is the starting period for which you want to calculate the depreciation. *Start_period* must use the same units as *life*.

**End_period** is the ending period for which you want to calculate the depreciation. *End_period* must use the same units as *life*.

**Factor** is the rate at which the balance declines. If factor is omitted, it is assumed to be 2 (the double-declining balance method). Change *factor* if you do not want to use the double-declining balance method. For a description of the double-declining balance method, see DDB earlier in this book.

**No_switch** is a logical value specifying whether to switch to straight-line depreciation when depreciation is greater than the declining balance calculation.

- If *no_switch* is TRUE, Microsoft Excel does not switch to straight-line depreciation even when the depreciation is greater than the declining balance calculation.
- If *no_switch* is FALSE or omitted, Microsoft Excel switches to straight-line depreciation when depreciation is greater than the declining balance calculation.

All arguments except *no_switch* must be positive numbers.

### Examples

Suppose a factory purchases a new machine. The machine costs $2,400 and has a lifetime of 10 years. The salvage value of the machine is $300. The following examples show depreciation over several periods. The results are rounded to two decimal places.

VDB(2400,300,3650,0,1) equals $1.32, the first day's depreciation. Microsoft Excel automatically assumes that *factor* is 2.

VDB(2400,300,120,0,1) equals $40.00, the first month's depreciation.

VDB(2400,300,10,0,1) equals $480.00, the first year's depreciation.

VDB(2400,300,120,6,18) equals $396.31, the depreciation between the 6th month and the 18th month.



VDB(2400,300,120,6,18,1.5) equals $311.81, the depreciation between the 6th month and the 18th month using a *factor* of 1.5 instead of the double-declining balance method.

Suppose instead that the $2,400 machine is purchased in the middle of the first quarter of the fiscal year. The following formula determines the amount of depreciation for the first fiscal year that you own the asset, assuming that tax laws limit you to 150% depreciation of the declining balance:

VDB(2400,300,10,0,0.875,1.5) equals $315.00

### Related Functions

| | |
|---|---|
| DDB | Returns the depreciation of an asset for a specified period using the double-declining balance method |
| SLN | Returns the straight-line depreciation of an asset for one period |
| SYD | Returns the sum-of-years' digits depreciation of an asset for a specified period |

*Macro Sheets Only*

## VIEW.3D(elevation,perspective,rotation,axes, height%)

## VIEW.3D?(elevation,perspective,rotation,axes, height%)

Equivalent to the Format 3-D View command. Adjusts the view of the active 3-D chart. Use VIEW.3D to emphasize different parts of your chart by viewing it from different angles and perspectives. For detailed information on the Format 3-D View command, see online Help.

**Elevation** is a number from –90 to 90 specifying the viewing elevation of the chart and is measured in degrees. *Elevation* corresponds to the Elevation box in the Format 3-D View dialog box.

- If *elevation* is 0, you view the chart straight on. If *elevation* is 90, you view the chart from above (a "bird's eye view"). If *elevation* is –90, you view the chart from below.
- If *elevation* is omitted, it is assumed to be 25.

**Perspective** is a number from 0 to 100 percent specifying the perspective of the chart. *Perspective* corresponds to the Perspective box in the Format 3-D View dialog box.

- A higher *perspective* value simulates a closer view.
- If *perspective* is omitted, it is assumed to be 30.

**Rotation** is a number from 0 to 360 specifying the rotation of the chart around the value (z) axis and is measured in degrees. *Rotation* corresponds to the Rotation box in the Format 3-D View dialog box. As you rotate the chart, the back and side walls are moved so that they do not block the chart. If *rotation* is omitted, it is assumed to be 30.

**Axes** is a logical value specifying whether axes are fixed in the plane of the screen or can rotate with the chart. *Axes* corresponds to the Right Angle Axes check box in the Format 3-D View dialog box.

- If *axes* is TRUE, Microsoft Excel locks the axes.
- If *axes* is FALSE, Microsoft Excel allows the axes to rotate.
- If *axes* is omitted and the chart view is 3-D layout, *axes* is assumed to be FALSE.
- If *axes* is omitted and the chart view is not 3-D layout, *axes* is assumed to be TRUE.

**Height%** is a number from 5 to 500 specifying the height of the chart as a percentage of the length of the base. *Height%* corresponds to the Height box in the Format 3-D View dialog box. *Height%* is useful for controlling the appearance of charts with many series or data points. If *height%* is omitted, it is assumed to be 100.

### Example

The following macro formula adjusts the selected 3-D chart so that its elevation angle is 45 degrees, it has a relatively close view of 75, it has no rotation, its axes are free to rotate, and its height is 150% of the length of the base:

VIEW.3D(45,75,0,FALSE,150)

### Related Function

| | |
|---|---|
| FORMAT.MAIN | Formats the main chart |

Macro Sheets Only

## VLINE(num_rows)

Scrolls the active window vertically by the number of rows you specify.

**Num_rows** is a number that specifies how many rows to scroll.

- If *num_rows* is positive, Microsoft Excel scrolls down by the number of rows indicated by *num_rows*.
- If *num_rows* is negative, Microsoft Excel scrolls up by the number of rows indicated by *num_rows*.

### Related Functions

| | |
|---|---|
| VPAGE | Vertically scrolls the active window one full window at a time |
| VSCROLL | Vertically scrolls a document by percentage or by row number |

## VLOOKUP(lookup_value,table_array, col_index_num)

Searches the leftmost column of an array for a particular value, and returns the value in the cell indicated. Use VLOOKUP instead of HLOOKUP when your comparison values are located in a column to the left or right of the data you want to find.

**Lookup_value** is the value to be found in the first column of the array. *Lookup_value* can be a value, a reference, or a quoted text string.

**Table_array** is the table of information in which data is looked up. Use a reference to a range or a range name, such as Database.

- The values in the first column of *table_array* must be placed in ascending order: ..., –2, –1, 0, 1, 2, ... , A–Z, FALSE, TRUE; otherwise VLOOKUP may not give the correct value.
- You can put the values in ascending order by choosing the Data Sort command and selecting Ascending.
- The values in the first column of *table_array* can be text, numbers, or logical values.
- Uppercase and lowercase text are equivalent.

**Col_index_num** is the column number in *table_array* from which the matching value should be returned. A *col_index_num* of 1 returns the value in the first column in *table_array*; a *col_index_num* of 2 returns the value in the second column in *table_array*, and so on. If *col_index_num* is less than 1, VLOOKUP returns the #VALUE! error value; if *col_index_num* is greater than the number of columns in *table_array*, VLOOKUP returns the #REF! error value.

### Notes

- If VLOOKUP can't find *lookup_value*, it uses the largest value that is less than or equal to *lookup_value*.
- If *lookup_value* is smaller than the smallest value in the first column of *table_array*, VLOOKUP returns the #N/A error value.

### Examples

| | A | B | C | D |
|---|---|---|---|---|
| 1 | Air at 1 atm pressure | | | |
| 2 | Density | Viscosity | Temp | |
| 3 | (kg/cubic m) | (kg/m*s)*1E+05 | (degrees C) | |
| 4 | 0.457 | 3.55 | 500 | |
| 5 | 0.525 | 3.25 | 400 | |
| 6 | 0.616 | 2.93 | 300 | |
| 7 | 0.675 | 2.75 | 250 | |
| 8 | 0.746 | 2.57 | 200 | |
| 9 | 0.835 | 2.38 | 150 | |
| 10 | 0.946 | 2.17 | 100 | |
| 11 | 1.09 | 1.95 | 50 | |
| 12 | 1.29 | 1.71 | 0 | |

In the worksheet above, where the range A4:C12 is named Range:

VLOOKUP(1,Range,1) equals 0.946

VLOOKUP(1,Range,2) equals 2.17

VLOOKUP(1,Range,3) equals 100

VLOOKUP(0.1,Range,2) equals #N/A, because 0.1 is less than the smallest value in column A

VLOOKUP(2,Range,2) equals 1.71



### Related Functions

| | |
|---|---|
| HLOOKUP | Looks in the first row of an array and moves down the column to return the value of a cell |
| INDEX | Uses an index to return a value from a reference or array |
| LOOKUP and MATCH | Look up values in an array or reference |

Macro Sheets Only

## VOLATILE( )

Specifies that a custom function is volatile. A volatile custom function is recalculated every time a calculation occurs on the worksheet.

### Notes

- VOLATILE must precede every other formula in the custom function except RESULT and ARGUMENT.
- Normally a worksheet recalculates a cell containing a nonvolatile custom function only when any part of the complete formula in the cell is recalculated.

### Example

The following volatile custom function generates random integers in the range you specify each time the worksheet is recalculated:

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **RandBetween** | |
| 4 | | =ARGUMENT("LowNumber",1) | Lower number |
| 5 | | =ARGUMENT("HighNumber",1) | Higher number |
| 6 | | =VOLATILE() | Makes function volatile |
| 7 | | LowNumber=INT(LowNumber) | Converts to an integer |
| 8 | | HighNumber=INT(HighNumber) | |
| 9 | RandInteger | =INT(RAND()*(HighNumber-LowNumber+1))+LowNumber | Computes a random integer |
| 10 | | =RETURN(RandInteger) | |

The complete formula in cell B9 is:

```
=INT(RAND()*(HighNumber-LowNumber+1))+LowNumber
```

The example for RAND shows the nonvolatile version of the preceding example.

### Related Function

| | |
|---|---|
| RESULT | Specifies the type of value a custom function returns |

Macro Sheets Only

## VPAGE(num_windows)

Vertically scrolls the active window one window at a time. Use VPAGE to change the displayed area of a worksheet or macro sheet.

**Num_windows** specifies the number of windows to scroll the active window vertically. A window is defined as the number of visible rows. If 20 rows are visible in the window, VPAGE scrolls in increments of 20 rows.

- If *num_windows* is positive, VPAGE scrolls down.
- If *num_windows* is negative, VPAGE scrolls up.

### Related Functions

| | |
|---|---|
| HPAGE | Horizontally scrolls the active window one full window at a time |
| HSCROLL | Horizontally scrolls a document by percentage or by column number |
| VSCROLL | Vertically scrolls a document by percentage or by row number |

Macro Sheets Only

## VSCROLL(position,row_logical)

Vertically scrolls the active document by percentage or by row number.

**Position** specifies the row you want to scroll to. *Position* can be an integer representing the row

number or a fraction or percentage representing the vertical position of the row in the document. If *position* is 0, VSCROLL scrolls your document to its top edge, which is row 1. If *position* is 1, VSCROLL scrolls your document to its bottom edge, which is row 16,384.

***Row_logical*** is a logical value specifying how the function scrolls.

- If *row_logical* is TRUE, VSCROLL scrolls the document to row *position*.
- If *row_logical* is FALSE or omitted, VSCROLL scrolls the document to the vertical position represented by the fraction *position*.

### Notes

- To scroll to a specific row *n*, either use VSCROLL(*n*,TRUE) or VSCROLL(*n*/16384). To scroll to row 138, for example, enter VSCROLL(138,TRUE) or VSCROLL(138/16384).
- If you are recording a macro and move the scroll box several times in a row, the recorder only records the final location of the scroll box, omitting any intermediate steps. Remember that scrolling does not change the active cell or the selection.

### Examples

The following macro formulas all scroll the document to row 8192, halfway down the worksheet:

```
VSCROLL(8192,TRUE)
VSCROLL(50%)
VSCROLL(0.5,FALSE)
VSCROLL(8192/16384)
```

The following macro goes to a cell, then positions the cell in the upper-left corner of the window:

| | B | C |
|---|---|---|
| 1 | *Formulas* | *Comments* |
| 2 | | |
| 3 | **PositionCell** | |
| 4 | =ACTIVATE("Worksheet1") | Activates a document |
| 5 | =FORMULA.GOTO("R400C14") | Goes to a cell |
| 6 | =VSCROLL(ROW(ACTIVE.CELL()),TRUE) | Puts active cell at window top |
| 7 | =HSCROLL(COLUMN(ACTIVE.CELL()),TRUE) | Puts active cell at window left |
| 8 | =RETURN() | |

### Related Functions

| | |
|---|---|
| FORMULA.GOTO | Selects the specified reference or named range |
| HPAGE | Horizontally scrolls the active window one full window at a time |
| HSCROLL | Horizontally scrolls a document by percentage or by column number |
| SELECT | Selects cells or objects on a sheet or an item on a chart |
| VPAGE | Vertically scrolls the active window one full window at a time |

*Macro Sheets Only*

## WAIT(serial_number)

Pauses the macro until the time specified by the serial number.

***Serial_number*** is the date-time code used by Microsoft Excel for date and time calculations. You can give *serial_number* as text, such as "4:30 PM", or as a formula, such as NOW()+"00:00:04", instead of as a number. The text or formula is automatically converted to a serial number. For more information about *serial_number*, see NOW.

### Important

WAIT suspends all Microsoft Excel activity and may prevent you from performing other operations on your computer. Background processes, such as printing and recalculation, are continued.

### Tip

You can use WAIT with NOW to pause a macro for a specified length of time. For example, the following macro formula waits 3 seconds from the time the functions are evaluated:

```
WAIT(NOW()+"00:00:03")
```

### Related Functions

| | |
|---|---|
| ON.TIME | Runs a macro at a specified time |
| NOW | Returns the serial number of the current date and time |



## WEEKDAY(serial_number)

Returns the day of the week corresponding to *serial_number*. The day is given as an integer, ranging from 1 (Sunday) to 7 (Saturday).

***Serial_number*** is the date-time code used by Microsoft Excel for date and time calculations. You can give *serial_number* as text, such as "15-Apr-1991" or "4-15-91", instead of as a number. The text is automatically converted to a serial number. For more information about *serial_number*, see NOW.

### Note

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

### Tips

- To display the result of the WEEKDAY function as a day of the week, use the Format Number command to format the cell containing the function with the custom date format "dddd".
- You can also use the TEXT function to convert a value to a specified number format:

  TEXT("4/16/90","dddd") equals Monday

### Examples

WEEKDAY("2/14/90") equals 4 (Wednesday)

If you are using the 1900 Date System (the default in Microsoft Excel for Windows or OS/2), then:

WEEKDAY(29747.007) equals 4 (Wednesday)

If you are using the 1904 Date System (the default in Microsoft Excel for the Macintosh), then:

WEEKDAY(29747.007) equals 3 (Tuesday)

### Related Functions

| | |
|---|---|
| DAY | Converts a serial number into a day |
| NOW | Returns the serial number of the current date and time |
| TEXT | Formats a number and converts it to text |
| TODAY | Returns the serial number of today's date |

*Macro Sheets Only*

## WHILE(logical_test)

Carries out the statements between the WHILE function and the next NEXT function until *logical_test* is FALSE. Use WHILE–NEXT loops to carry out a series of macro formulas while a certain condition remains TRUE.

***Logical_test*** is a value or formula that evaluates to TRUE or FALSE. If *logical_test* is FALSE the first time the WHILE function is reached, the macro skips the loop and resumes running at the statement after the next NEXT function. If there is no NEXT function in the same column, WHILE displays an error message and interrupts the macro.

For information on WHILE–NEXT loops, see Chapter 19, "Designing and Writing a Command Macro," in the *Microsoft Excel User's Guide*.

### Example

The following macro illustrates how to use a WHILE–NEXT loop to enclose functions that you want to run repeatedly until a certain condition is met.

| | A | B | C |
|---|---|---|---|
| 1 | *Names* | *Formulas* | *Comments* |
| 2 | | | |
| 3 | | **ExitLoop** | |
| 4 | | =SET.VALUE(Counter,1) | Initializes Counter to 1 |
| 5 | | =WHILE(Counter<10) | Begins WHILE–NEXT loop |
| 6 | Counter | =Counter+INT(RAND()*5) | Increments Counter |
| 7 | | =ProcessCounter(Counter) | Runs subroutine |
| 9 | | =IF(Counter=8,BREAK()) | If Counter = 8, break |
| 9 | | =NEXT() | Ends WHILE–NEXT loop |
| 10 | | =RETURN() | Ends the macro |

### Related Functions

| | |
|---|---|
| FOR | Begins a FOR–NEXT or FOR–WHILE loop |
| FOR.CELL | Begins a FOR.CELL–NEXT or FOR.CELL–WHILE loop |
| IF | Specifies a logical test to perform and precedes the block of functions that will be carried out if the logical test is TRUE |
| NEXT | Ends a FOR–NEXT, FOR.CELL–NEXT, or WHILE–NEXT loop |

Macro Sheets Only

## WINDOWS(*type_num*)

Returns the names of all open Microsoft Excel windows, including hidden windows. Use WINDOWS to get a list of all active windows for use by other macro functions that return information about or manipulate windows, such as GET.WINDOW and ACTIVATE.

The names are returned as a horizontal array of text values, in order of their appearance on your screen. The first name is the active window, the next name is the window directly under the active window, and so on.

***Type_num*** is a number that specifies which types of documents are returned by WINDOWS.

| *Type_num* | Returns window names from these types of documents |
|---|---|
| 1 or omitted | All windows except those belonging to add-in documents |
| 2 | Add-in documents only |
| 3 | All types of documents |

### Tips

- You can change the output of a horizontal array to vertical with the TRANSPOSE function.
- You can use WINDOWS with the INDEX function to choose individual window names from the array for use in other functions that take window names as arguments.
- You can use the COLUMNS functions to count the number of entries in the array.

### Examples

If the active window, named Sheet1, is on top of a window named Macro1:2, which is on top of a window named Macro1:1, then:

WINDOWS( ) equals
{"Sheet1","Macro1:2","Macro1:1"}

For another example of WINDOWS, see UNHIDE.

### Related Functions

| | |
|---|---|
| ACTIVATE | Activates a window or a pane of a window |
| DOCUMENTS | Returns the names of all open documents |
| GET.WINDOW | Returns information about a window |
| NEW.WINDOW | Creates a new window for an existing worksheet or macro sheet |
| ON.WINDOW | Runs a macro when the specified window is activated |

Macro Sheets Only

## WORKGROUP(*name_array*)

Equivalent to the Window Workgroup command. Creates a workgroup. For detailed information on the Window Workgroup command, see online Help.

***Name_array*** is the list of open, unhidden worksheets and macro sheets, as text, that you want to include in a workgroup. Charts cannot be part of a workgroup. If *name_array* is omitted, the most recently created workgroup is recreated. If no workgroup has been created during the current Microsoft Excel session, all open, unhidden worksheets are created as a workgroup.

WORKGROUP returns the #VALUE! error value and interrupts the macro if it can't find any of the documents in *name_array* or if any of the documents is a chart.

### Examples

For Microsoft Excel for Windows or OS/2, the following macro formula creates a workgroup from the specified worksheets:

WORKGROUP({"SALES.XLS","EXPENSES.XLS", "PROFITS.XLS"})



In Microsoft Excel for the Macintosh, the following macro formula creates a workgroup from the specified worksheets:

WORKGROUP({"SALES","EXPENSES","PROFITS"})

### Related Function

| | |
|---|---|
| FILL.WORKGROUP | Fills the same range in all workgroup sheets with the selection |

Macro Sheets Only

## WORKSPACE(*fixed,decimals,r1c1,scroll, status,formula,menu_key,remote,entermove, underlines,tools,notes,nav_keys,menu_key_action*)
## WORKSPACE?(*fixed,decimals,r1c1,scroll, status,formula,menu_key,remote,entermove, underlines,tools,notes,nav_keys,menu_key_action*)

Equivalent to the Options Workspace command (full menus). Use WORKSPACE to change the workspace settings for a document. For detailed information on the Options Workspace command, see online Help.

Arguments correspond to check boxes and text boxes in the Options Workspace dialog box. Arguments corresponding to check boxes are logical values. If an argument is TRUE, the check box is turned on; if FALSE, the check box is turned off.

***Fixed*** corresponds to the Fixed Decimal check box.

***Decimals*** specifies the number of decimal places. *Decimals* is ignored if *fixed* is FALSE or omitted.

***R1c1*** corresponds to the R1C1 check box.

***Scroll*** corresponds to the Scroll Bars check box.

***Status*** corresponds to the Status Bar check box.

***Formula*** corresponds to the Formula Bar check box.

***Menu_key*** is a text value indicating an alternate menu key, and corresponds to the Alternate Menu Or Help Key box.

***Remote*** corresponds to the Ignore Remote Requests check box. In Microsoft Excel for the Macintosh, *remote* is only available if you are using system software version 7.0 or later.

***Entermove*** corresponds to the Move Selection After Enter/Return check box.

***Underlines*** is a number corresponding to the Command Underline options as shown in the following table. *Underlines* is only available in Microsoft Excel for the Macintosh.

| *Underlines* | Command underlines are |
|---|---|
| 1 | On |
| 2 | Off |
| 3 | Automatic |

***Tools*** corresponds to the Tool Bar check box.

***Notes*** corresponds to the Note Indicator check box.

***Nav_keys*** corresponds to the Alternate Navigation Keys check box. In Microsoft Excel for the Macintosh, *nav_keys* is ignored.

***Menu_key_action*** is the number 1 or 2 specifying options for the alternate menu or Help key. In Microsoft Excel for the Macintosh, *menu_key_action* is ignored.

| *Menu_key_action* | Alternate menu or Help key activates |
|---|---|
| 1 or omitted | Microsoft Excel menus |
| 2 | Lotus 1-2-3 Help |

### Related Function

| | |
|---|---|
| GET.WORKSPACE | Returns information about the workspace |

## YEAR(*serial_number*)

Returns the year corresponding to *serial_number*. The year is given as an integer in the range 1900–2078.

***Serial_number*** is the date-time code used by Microsoft Excel for date and time calculations. You can give *serial_number* as text, such as "15-Apr-1991" or "4-15-91", instead of as a number. The text is automatically converted to a serial number. For more information about *serial_number*, see NOW.

#### Note

Microsoft Excel for Windows or OS/2 and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

#### Examples

YEAR("7/5/90") equals 1990

If you are using the 1900 Date System (the default in Microsoft Excel for Windows or OS/2), then:

YEAR(0.007) equals 1900

YEAR(29747.007) equals 1981

If you are using the 1904 Date System (the default in Microsoft Excel for the Macintosh), then:

YEAR(0.007) equals 1904

YEAR(29747.007) equals 1985

#### Related Functions

| | |
|---|---|
| MONTH, DAY, WEEKDAY, HOUR, MINUTE, and SECOND | Convert serial numbers into months, days, days of the week, hours, minutes, and seconds |
| NOW | Returns the serial number of the current date and time |
| TODAY | Returns the serial number of today's date |

# Appendix A Using the CALL and REGISTER Functions

The following tables are for use with the CALL and REGISTER functions. Arguments and return values differ slightly depending on your operating environment; refer to either "Data Types in Microsoft Excel for Windows or OS/2" or "Data Types in Microsoft Excel for the Macintosh" below. The letters A to P are defined as codes for different data types and are used to form *type_text* (the third argument to REGISTER). The first letter of *type_text* specifies the return value, and subsequent letters specify the data type of each argument. See the examples for more information on specifying *type_text*.

## Data Types in Microsoft Excel for Windows or OS/2

#### Important

In Microsoft Excel version 3.0, the value returned by code B is a floating-point number; in earlier versions of Microsoft Excel, code B returns a pointer to a value. If you have code resources (also known as dynamic link libraries, or DLL) that worked with earlier versions of Microsoft Excel, you must change code B to code E in the macro function that registers the code resource. For example, the formula REGISTER("MATH.DLL","BESSEL","BIB") should be changed to REGISTER("MATH.DLL","BESSEL","EIB").

| Code | Data type | Argument pushed on the stack | Value returned by code resource |
|---|---|---|---|
| A | Boolean, passed by value. | FALSE = 0; TRUE = 1. Two-byte integer is pushed. | FALSE if 0; otherwise TRUE. |
| B | IEEE float, passed by value. | IEEE floating-point number (8 bytes). | IEEE floating-point number. |
| C | Zero-terminated ANSI string, passed by reference. | Far pointer to zero-terminated string. | Far pointer to zero-terminated string. |
| D | Byte-counted ANSI string, passed by reference. | Far pointer to a byte that is followed by a string. The byte gives the number of characters in the string. There is no termination character at the end of the string. | Far pointer to a string with the first byte holding the count of the following characters. |
| E | IEEE float buffer, passed by reference. | Far pointer to a four-word (8 byte) number buffer. Buffer is filled with the number provided as the argument to CALL. | Far pointer to IEEE floating-point number. |
| F | String buffer for a zero-terminated ANSI string, passed by reference. | Far pointer to a 256-byte string buffer. Buffer is filled with the zero-terminated string provided as the argument to CALL. | Zero-terminated contents of the string buffer. For more information, see "Special Data Types" below. |

| Code | Data type | Argument pushed on the stack | Value returned by code resource |
|---|---|---|---|
| G | String buffer for a byte-counted ANSI string, passed by reference. | Far pointer to a 256-byte string buffer. Buffer is filled with the byte-counted string provided as the argument to CALL. | Byte-counted contents of the string buffer. For more information, see "Special Data Types" below. |
| H | Unsigned integer, passed by value. | One word (2 bytes). | Integer. |
| I | Signed integer, passed by value. | One word (2 bytes). | Signed integer. |
| J | Signed long, passed by value. | Two words (4 bytes). | Signed long integer. |
| K | Floating-point array, passed by reference. | Far pointer to array structure. The first word of the structure contains number of rows. The next word contains number of columns. Followed by row × columns IEEE floating-point numbers. | Far pointer to the array structure. |
| L | Boolean, passed by reference. | Far pointer to a 2-byte integer buffer. FALSE = 0; TRUE = 1. | Far pointer to a 2-byte integer. FALSE if 0; otherwise TRUE. |
| M | Signed short, passed by reference. | Far pointer to a 2-byte integer buffer. | Far pointer to a signed integer. |
| N | Signed long, passed by reference. | Far pointer to a 4-byte integer buffer. | Far pointer to a signed long integer. |
| O | Floating-point array, passed by reference. | Three arguments are pushed:<br>1. Two-byte integer indicating the number of rows.<br>2. Two-byte signed integer indicating the number of columns.<br>3. Far pointer to a 2-D array of IEEE floating-point numbers. | Not allowed. |
| P | General, passed by reference. | Far pointer to the Oper Data Structure. For more information, see "Oper Data Structures" below. | Far pointer to the Oper Data Structure. |



## *Data Types in Microsoft Excel for the Macintosh*

| Code | Data type | Argument pushed on the stack | Value returned by code resource |
|---|---|---|---|
| A | Boolean, passed by value. | FALSE = 0; TRUE = 1. Two-byte integer is pushed. | FALSE if 0; otherwise TRUE. |
| B | IEEE float, passed by reference. | Pointer to IEEE floating-point number. | Pointer to IEEE floating-point number. |
| C | Zero-terminated ANSI string, passed by reference. | Pointer to zero-terminated string. | Pointer to zero-terminated string. |
| D | Byte-counted ANSI string, passed by reference. | Pointer to a byte that is followed by a string. The byte gives the number of characters in the string. There is no termination character at the end of the string. | Pointer to a string with the first byte holding the count of the following characters. |
| E | IEEE float buffer, passed by reference. | Pointer to a four-word (8 byte) number buffer. Buffer is filled with the number provided as the argument to CALL. | Pointer to IEEE floating-point number. |
| F | String buffer for zero-terminated ANSI string, passed by reference. | Pointer to a 256-byte string buffer. Buffer is filled with the zero-terminated string provided as the argument to CALL. | Zero-terminated contents of the string buffer. For more information, see "Special Data Types" below. |
| G | String buffer for byte-counted ANSI string, passed by reference. | Pointer to a 256-byte string buffer. Buffer is filled with the byte-counted string provided as the argument to CALL. | Byte-counted contents of the string buffer. For more information, see "Special Data Types" below. |
| H | Unsigned integer, passed by value. | One word (2 bytes). | Integer. |
| I | Signed integer, passed by value. | One word (2 bytes). | Signed integer. |
| J | Signed long, passed by value. | Two words (4 bytes). | Signed long integer. |
| K | Floating-point array, passed by reference. | Pointer to array structure. The first word of the structure contains number of rows. The next word contains number of columns. Followed by row × columns IEEE floating-point numbers. | Pointer to the array structure. |
| L | Boolean, passed by reference. | Pointer to a 2-byte integer buffer. FALSE = 0; TRUE = 1. | Pointer to a 2-byte integer. FALSE if 0; otherwise TRUE. |
| M | Signed short, passed by reference. | Pointer to a 2-byte integer buffer. | Pointer to a signed integer. |

| Code | Data type | Argument pushed on the stack | Value returned by code resource |
|---|---|---|---|
| N | Signed long, passed by reference. | Pointer to a 4-byte integer buffer. | Pointer to a signed long integer. |
| O | Floating-point array, passed by reference. | Three arguments are pushed:<br>1. Signed short integer indicating the number of rows.<br>2. Signed short integer indicating the number of columns.<br>3. Pointer to a 2-D array of IEEE floating-point numbers. | Not allowed. |
| P | General, passed by reference. | Pointer to the Oper Data Structure. For more information, see "Oper Data Structures" below. | Pointer to the Oper Data Structure. |

### *Special Data Types*

- Code-resource functions that don't return a value (for example, VOID in the C language or SUBROUTINE in Fortran) must use ">" to give Microsoft Excel the return value it requires. Use ">" as the first letter of *type_text* to specify the next letter of *type_text* as the value to be returned to Microsoft Excel. The code resource modifies the first argument and uses it as the return value. Only data types passed by reference may be modified by the code resource. Codes A, H, I, and J are passed by value and cannot be modified by the code resource. In these cases, the passed value is also used as the return value. For compatibility reasons, code B is parsed by all versions of Microsoft Excel as though it were passed by value, and cannot be modified by the code resource.
- If the return data type is F or G, then the return value from the code-resource function is ignored. Instead, Microsoft Excel searches the list of function arguments for the first corresponding data type (F or G) and takes the current contents of the allocated string buffer as the return value for the CALL function.
- To make a code-resource function volatile, use "!" as the last character in the *type_text* argument. Volatile functions are recalculated each time the worksheet is recalculated.

### *Notes*

- REGISTER uses the standard Pascal calling convention for passing arguments to procedures and returning values.
- Codes E, F, and G are pointers to buffers. In earlier versions of Microsoft Excel, these buffers could only be used to return values from the called procedure. In Microsoft Excel version 3.0, the buffers are filled with the argument provided to the CALL function. This is useful when passing arguments by reference rather than by value.
- In earlier versions of Microsoft Excel, arguments to called procedures were not coerced. In Microsoft Excel version 3.0, arguments are coerced the same way that arguments to other functions are coerced; that is, if instead of a number you give a reference to a cell that contains the number or a name defined as that number, the number value will be used.



### *Examples*

The *type_text* value "AEEI" means that the procedure returns a boolean value (A) and takes three arguments. The first two arguments are pointers to IEEE floating-point numbers (EE) and the third argument is a signed integer (I). Two words (4 bytes) are pushed for each of the first two arguments; one word (2 bytes) is pushed for the third argument. The return value is FALSE if the function returns 0, and TRUE if anything else is returned.

If the argument type is ">C", Microsoft Excel allocates a 256-byte buffer, copies the string provided as the worksheet-function argument to the buffer, and passes a pointer to the buffer as an argument to the code-resource function. The code-resource function may modify the content of the buffer and may return without returning a value. The value returned by the worksheet function is the content of the buffer when the code-resource function returns.

### *Oper Data Structures*

Using code P as part of the *type_text* argument to the REGISTER function allows you to send and return any Microsoft Excel data type, which can then be identified through the last 2 bytes of the data type. With code types A to O you must specify a particular Microsoft Excel data type for each argument. The table below lists data types that can be passed by oper data structures. The first 8 bytes contain the value of the object; the last 2 bytes identify the type of object, as follows:

| Oper type | Two-byte object identifier | Contents of first 8 bytes (value) |
|---|---|---|
| Double | 1 | Eight-byte floating-point value. |
| String | 2 | Four bytes with a far pointer to the string. The string is byte-counted. The remaining 4 bytes are unused. |
| Boolean | 4 | Two-byte boolean value (0 = FALSE, 1 = TRUE), followed by 6 unused bytes. |
| Error | 16 | Two-byte error value. Error values are:<br>0 = #NULL!<br>7 = #DIV/0!<br>15 = #VAL!<br>23 = #REF!<br>29 = #NAME?<br>36 = #NUM!<br>42 = #N/A<br>The remaining 6 bytes are unused. |
| Array | 64 | Four bytes with a far pointer to an array of oper data structures, followed by 2 bytes containing the number of rows and 2 bytes containing the number of columns. The elements of the array can be any data type except arrays. |

The next two arguments are used only when arguments are passed to the code resource. They may not be used for return values.

| Oper type | Two-byte object identifier | Contents of first 8 bytes (value) |
|---|---|---|
| Missing arg | 128 | Unused. This oper type is used only when the corresponding argument to the called function is omitted. For example, when the actual number of arguments is less than the number specified by the REGISTER function, the missing argument identifier is sent. |
| Empty cell | 256 | Unused. This oper type is used only when one of the arguments is a reference to an empty cell. |

# Index


3-D charts 82, 83, 245
(&) *See* Ampersand
(*) *See* Asterisk
(/) *See* Backslash
(^) *See* Caret
(,) *See* Comma
(!) *See* Exclamation point
(?) *See* Question mark
(') *See* Quotation marks
(") *See* Quotation marks

## A

A1 mode 117
A1 reference style
  cell address 7
  converting text to xxvi, 235-236
  converting to R1C1 style xxiii, 91
  information, returning 117
  recording macros 203
  specifying xiii, 1
A1.R1C1 function xiii, 1
ABS function xviii, 1
Absolute reference
  cell address 7
  converting to
    ABSREF function 1
    automatic conversion 54
    DEREF function xxv, 54
    FORMULA.CONVERT function xxiii, 91
    REFTEXT function xxvi, 1, 3, 192-193, 214
  defining names 217
  relative, converting to
    FORMULA.CONVERT function xxiii, 91
    RELREF function 194-195
  returning xxv, 1-2, 103
Absolute value xviii, 1
ABSREF function xxv, 1-2
ACOS function xx, 2
ACOSH function xx, 2
ACTIVATE function xxiii, 2-3, 122, 224
ACTIVATE.NEXT function xiii, 3
ACTIVATE.PREV function xiii, 3
Active cell
  changing xiii, 209-210
  displaying 222
  moving to unlocked cell 241
  returning reference 3-4
ACTIVE.CELL function xv, 3-4, 110
Add-in macro file format 204
ADD.ARROW function x, 4
ADD.BAR function
  deleting custom menu bars 51
  description xiii, xiv, 4-5
  menu bar ID number 67, 195, 222
ADD.COMMAND function xiii, 5-6, 51, 195
ADD.MENU function xiii, 5, 6-7
ADD.OVERLAY function x, 7
Adding
  array products viii, 232
  DSUM function xvi, 62
  SUM function xviii, 231
  SUMPRODUCT function xviii, 232
  while pasting 177
ADDRESS function xvii, 7-8
ALERT function xiv, 8-9, 125, 151
Alert message
  *See also* Status bar
  custom xiv, 8-9
  sounding tone xxiii, 21
Alignment
  aligning cell contents 9
  defining styles 50
  formatting text 88
  information, returning 27, 104, 113
  types 9
ALIGNMENT function xxi, 9
ALT key combinations
  ALT, SPACEBAR, M, moving active window 12
  ALT, SPACEBAR, S, sizing windows 13
  assigning to macros 165-166
Ampersand (&), command access key notation
  custom command 103
  renamed command 195
AND function xiii, xviii, 9-10
Angle
  arccosine 2
  arcsine 16
  arctangent 18
  cosine 37
  pie chart slice 86
  sine 225
  tangent 234
Annuity
  *See also* Investment
  defined 190
  functions 190
  interest rate *See* Interest
  payments *See* Payment
ANSI character set 29, 32
APP.ACTIVATE function xxiii, xxvi, 10-11
APP.MAXIMIZE function xx, xxvi, 11
APP.MINIMIZE function xx, xxvi, 11
APP.MOVE function xx, xxvi, 11-12
APP.MOVE? function 12
APP.RESTORE function xx, xxvi, 12
APP.SIZE function xx, xxvi, 12-13
APP.SIZE? function 13
Application, external
  activating xxiii, 10-11
  carrying out commands xiv, 71-72
  closing DDE channel xxv, 234
  disabling external input 58
  importing text 30
  linking documents, pasting links 176
  requesting data xxv, 197-198
  running in background 70
  running macros xxv, 164-165
  sending data to xxv, 183
  sending keystrokes to xxv, 214-215
  starting xiv, 70-71
  task ID number 70, 130
  window 70
APPLY.NAMES function xxii, 13-14
APPLY.STYLE function xxi, 14, 228
Arc
  drawing 40
  moving 84
  object type 112

Arccosine xx, 2
Arcsine xx, 16
Arctangent xx, 17, 18
Area
  defined 14
  returning number in reference xvii, 14-15
Area chart
  chart type 82, 85
  creating 100-101
  drop lines 86
  formatting 83
  hi-lo lines *See* Hi-lo lines
AREAS function xvii, 14-15
Argument
  command-equivalent functions x
  converting data types xiv-xv
  custom functions xiv, 15-16
  data types xii-xiv, 15
  listing xv
  multiple references 14
  notational conventions xii
  optional xi
  overview xii
  required xi
  reversing logic xviii
  syntax xii
ARGUMENT function xiv, 15-16
ARRANGE.ALL function xxii, 16, 159
Array
  entering 129
  formulas
    entering xxiii, 90
    evaluating xiv
  index number 30
  information, returning 33, 34
  looking up values
    HLOOKUP function xviii, 12
    INDEX function xii, 127-129
    LOOKUP function xviii, 146-147
    MATCH function xviii, 148
    VLOOKUP function xviii, 246-247
Array, *continued*
  matrix
    determinant xviii, 149-150
    inverse matrix xviii, 153-154
    product of arrays xviii, 155, 232
  multiplying numbers 187
  naming 217
  orientation, changing 75
  relative position, returning 148
  returning to macro sheet 217
  returning values 128-129
  selecting values 30
  testing value type 136
  transposing xviii, 158, 237-238
  using in arguments xiii-xiv
Arrow
  adding to chart xiv, 4
  arrowhead, returning information 114
  deleting from chart xx, 51
  formatting 4, 179, 180
  moving 85
  positioning 4
  selecting 51
  sizing 4, 88
ARROW keys 43
ASIN function xx, 16
ASINH function xx, 17
Asset
  depreciation *See* Depreciation
  salvage value 47
  useful life 47
ASSIGN.TO.OBJECT function xxii, 17, 41
Asterisk (*) notation
  deleting files 74, 75
  opening files 169
  replacing text 94
  returning filenames 75
  searching text 208
ATAN function xx, 17, 18
ATAN2 function xx, 18
ATANH function xx, 18
ATTACH.TEXT function xx, 18-19
ATTRIBUTES function xxvii, 19-20
Auto-sum button 231
Auto_Close macro 120, 191
Auto_Open macro 120
Automatic background 81
Autoshared option 19
AVERAGE function xix, 20, 35, 193
Averaging
  AVERAGE function xxv, 20, 35, 193
  DAVERAGE function xvi, 46, 56
AXES function xx, 21
Axis
  attaching text 18
  displaying in chart 20-21
  formatting 81
  scaling 206-208


## B


Background, cell
  color 114, 180
  information, returning 105
  types 81
Background processes 248
Backslash (/), fill-aligned text 27
BACKSPACE key code 166
Backup file 203, 204
Bar chart
  chart type 82, 85
  creating 101
  formatting 83
  gap width 86
  overlap 86
Beep *See* Tone
BEEP function xxiii, 21
Blank cell
  returning information about xvii, 135
  skipping 80
  value 20
Bold formatting
  formatting font 81, 230
  returning cell information 104
  returning default font information 108





Border
  adding 21-22
  defining styles 49, 50
  formatting 114, 180
  line types 21, 114
  returning information about 104
  sizing 88
  styles 114
BORDER function xxi, 22
Bottom margin 174
BREAK function xiv, 22, 79
BREAK key code 166
BRING.TO.FRONT function xxi, 22
Buffer 256, 257
Button 40, 112


## C


C language 256
CALCULATE.DOCUMENT function xxii, 23
CALCULATE.NOW function xx, xii, 23
Calculating
  active document 23
  all open documents 23
  automatically 23
  background process 248
  before saving 24
  controlling 23-24
  excepting tables 23
  lines 138, 141
  manually 23
  mode, returning 130, 180
  nonvolatile custom functions 247
  ON.DATA function 165
  random numbers 191
  running macro xxv, 167
  speed 185
  stored vs. displayed values 23
  testing outcome 136
  types 23-24
  using names in functions xiii
  volatile functions xxv, 247, 256
  while pasting 177
CALCULATION function xxii, 23-24
CALL function
  buffers 256
  data types
    Microsoft Excel for the Macintosh 255-256
    Microsoft Excel for Windows or OS/2 253-254
  description xxiv 24
  unregistered functions 193, 194
CALLER function xxv, 24
Calling
  caller cell information xxv, 24
  external procedures xxiv
  REGISTER function 256
  unregistered functions 193, 194
Cancel button
  dialog-box definition 57
  displaying 131
  interrupting macros 8
CANCEL.COPY function xxiii, 25
CANCEL.KEY function xiv, 25-26
Capitalization *See* Uppercase characters
CAPS LOCK key code 166
Caret (^) notation
  centered text 27
  exponentiation operator 72
  key code 166
Case *See* Lowercase characters; Uppercase characters
Cash flow *See* Investment
Category axis
  attaching text 18, 19
  scaling 206, 207
Category label 64
Cell
  absolute reference, returning 103
  active *See* Active cell
  background color 180
  border *See* Border
  clearing 31
  empty *See* Blank cell
  filling 76, 197
  formatting *See* Formatting
  hiding 28
  information, returning xvii, 26, 27, 103-105
Cell, *continued*
  locking *See* Locking
  nonblank *See* Nonblank cell
  note *See* Note
  pasting 133
  pattern 104, 180
  protection *See* Locking
  reference *See* Reference
  selecting xxiii, 209-210, 212-214
  shifting after deletion 63
  style *See* Style
  type 104
  value xxv, 220, 221
CELL function xvii, 26-27
CELL.PROTECTION function xxi, 28
Centering 174
CHANGE.LINK function xxi, 28-29
Channel *See* Application, external
CHAR function xix, 29
Character
  capitalization *See* Uppercase characters
  code *See* Character set
  nonprintable xix, 30
  number in file xiv, 98
  replacing *See* Replacing
  returning last in string 200
  separator 140, 145
  wildcard *See* Wildcard characters
Character set
  returning numeric character code 42
  translating character codes 29
  types 29
Chart
  arrow *See* Arrow
  attaching text 18-19
  axis
    displaying 20-21
    scaling 206-208
  bar chart *See* Bar chart
  category labels 64
  clearing 31
  column chart *See* Column chart
  copying 36
  creating xxi, 159

Chart, *continued*
data series 64, 216-217, 119
drop lines *See* Drop lines
embedded 40, 112
formatting
combination chart 35
data points 86
default, changing 218
font 81-82
FORMAT.MAIN function 82-84
gridlines 119
legend 82
patterns 179
text 88
hi-lo lines *See* Hi-lo lines
information, returning xxvi, 105-106, 108
legend *See* Legend
line chart *See* Line chart
moving items 84-85
overlay *See* Overlay chart
pie *See* Pie chart
printing 174
protecting *See* Protecting documents
scatter chart 102
selecting items 182, 211-212, 213
size 42, 87-88
stacking pictures *See* Picture chart
text label
attaching 18-19
formatting 88-89
moving 84-85
type, returning 108
view 82-83
xy (scatter) chart 102
Chart Add Arrow command xx, 4
Chart Add Legend command xx, 137
Chart Add Overlay command xx, 7
Chart Attach Text command xx, 18, 88
Chart Axes command xx, 20
Chart Calculate Now command xx, 23
Chart Delete Arrow command xx, 51
Chart Delete Legend command 137
Chart Delete Overlay command xx, 53
Chart Edit Series command xx, 64
Chart Full Menus command 222
Chart Gridlines command xx, 119
Chart Main Chart Type command 147
Chart menu bar 5, 222
Chart Overlay Chart Type command 173
Chart Select Chart command xx, 212
Chart Select Plot Area command xx, 213
Chart Short Menus command 222
Check box
dialog-box definition 57
macro function arguments x
Check mark *See* Command
CHECK.COMMAND function xiv, 29
CHOOSE function xviii, 30
Circle, calculating area 182
CLEAN function xix, 30
CLEAR function xx, 31
CLEAR key code 166
Clipboard
closing window 223
copying picture to 37
displaying contents 223
moving 157
objects, shared view 221
CLOSE function xx, 31-32, 223
CLOSE.ALL function xxi, 32
Closing
active window 31-32
all open windows 32
files xiv, 73
quitting Microsoft Excel 191
CODE function xix, 32
Code Page 850 character set 29, 32
Code resource
C language 256
calling procedures from xxiv
compatibility across operating systems 253-254
data types 253-254
Fortran 256
making volatile 256
registering xxv, 24, 193-194
unregistering xxv, 242
Code, character *See* Character set
Color
background 105, 114, 180
borders 22
chart data points 83, 86
copying color palettes 32-33
defining 62-63
fill 114
font
formatting font 81
returning cell information 104
returning default font information 108
foreground 180
gridlines 115
headings 115
number format 26, 27
printing 186
screen display 59-60
COLOR.PALETTE function xxii, 32-33
Column
counting in array xvii, 34, 116
deleting 63
filling 76
freezing titles 97
headings
displaying 59, 115
printing 174
reference style xxiii, 1
hiding 34
information, returning
first used column 108
last used column 108
number xvii, 26, 201
width 104
outline
demoting xxiii, 54
displaying levels 224
expanding 223, 224
promoting xxiii, 187, 188
scrolling by column xxiii, 124-125
sorting 226-227
width, setting 33-34
Column chart
chart type 82, 85
creating 101
formatting 82-86





COLUMN function xvii, 33
COLUMN.WIDTH function xxi, 33-34
COLUMNS function xvii, 34, 60, 75, 158, 250
COMBINATION function xxii, 35
Comma (,) notation
listing arguments xv
union operator xv
Command
adding to menu xxiii, 5-6, 103
check mark, adding/removing xiv, 29
custom
access key notation 103
creating x
custom Help topics 6
enabling/disabling xiv, 67-68
macro sheet reference 5
dimmed 67
external applications *See* Application, external
ID number 51
macro function equivalents xx-xxii
menu position 5, 67, 103
removing xiv, 51-52
renaming xiv, 195
repeating 64
restoring deleted command 5
returning menu name 103
selection, automating ix, x
underlined letter 195, 251
undoing 241
using name in functions 5
COMMAND key combinations
assigning to macros 165
COMMAND+DELETE, scrolling to active cell 222
COMMAND+DOWN ARROW, selecting bottom cell 212
COMMAND+ENTER, entering arrays 69, 90, 129
COMMAND+H, finding next formula 93
COMMAND+LEFT ARROW, selecting leftmost cell 212
COMMAND+M, activating next window 3
COMMAND key combinations, *continued*
COMMAND+PERIOD
canceling marquee 25
interrupting macros 25-26
COMMAND+RIGHT ARROW, selecting rightmost cell 212
COMMAND+SHIFT+H, finding previous formula 93
COMMAND+SHIFT+M, activating previous window 3
COMMAND+UP ARROW, selecting topmost cell 212
Command macro *See* Macro
CONSOLIDATE function xx, 35-36, 109
Consolidating data
by category 109
consolidation functions 35-36
returning function used 109
returning list of references 109
Control Close command xx, 31
CONTROL key combinations *See* CTRL key combinations
Control macro functions xiv, xxv
Control Maximize command xx, 11
Control Minimize command xx, 11
Control Move command xx, 11, 156
Control Restore command xx, 12
Control Size command xx, 12, 226
Control Split command 227
COPY function xx, 36
Copy mode 117
COPY.CHART function xx, 36
COPY.PICTURE function xxi, 37
Copying
cell contents to fill selection 76
charts 36
color palette 32-33
COPY function 36
formats 76
graphic selection 37
objects 19
COS function xx, 37
COSH function xx, 37-38
Cosine xx, 2, 37-38
COUNT function xix, 35, 38
COUNTA function xiii, xix, 35, 38-39
Counter 79
Counting
characters xix, 137
columns 34
entries 75
nonblank cells xvi
numbers
COUNT function xix, 38
COUNTA function xix, 38-39
DCOUNT function xvi, 47
DCOUNTA function xvi, 47
rows 116, 202
Country settings 140
CREATE.NAMES function xxii, 39
CREATE.OBJECT function xxiii, 40-41, 122, 211
CREATE.PUBLISHER function xxi, xxvii, 41-42, 65
Criteria
defining 217
description 55
CSV file format 204
CTRL key combinations
assigning to macros 165
CTRL+BACKSPACE, scrolling to active cell 222
CTRL+COMMAND, scrolling to active cell 222
CTRL+DOWN ARROW, selecting bottom cell 212
CTRL+F5, returning window to previous size 98
CTRL+F6, activating next window 3
CTRL+F10, restoring window to full size 98
CTRL+LEFT ARROW, selecting leftmost cell 212
CTRL+RIGHT ARROW, selecting rightmost cell 212
CTRL+SHIFT+ENTER, entering formula as array 69, 90, 169
CTRL+SHIFT+F6, activating previous window 3
CTRL+SHIFT+F7, finding previous formula 92
CTRL+UP ARROW, selecting topmost cell 212
Currency values 27, 199
Curve, exponential 119, 120, 144-145
CUSTOM.REPEAT function xiv, 42

CUSTOM.UNDO function xiv, 42-43
Customizing functions xxiii-xxiv
CUT function xxi, 43
Cut mode 117

## D

Data
consolidating 35-36, 109
exponential curve 144, 145
oper data structures 257
parsing 175-176
series *See* Data series
sorting 226-227
types *See* Data types
Data Consolidate command xx, 35
Data Delete command xx, 43
Data Exit Find command 43
Data Extract command xx, 72
Data Find command xx, 43
Data Find mode 117
Data form 43, 44
Data Form command xx, 43
Data label
font 81-82
formatting 89
Data markers 180
Data Parse command xx, 175
Data point 18, 31
Data series
clearing 31
deleting 64
distribution method 86
editing 216
in chart 216-217
labeling 18
modifying 64, 65
plot order 64
Data Series command xx, 44
Data Set Criteria command xx, 55, 217
Data Set Database command xx, 55, 217
Data Set Extract command xx, 217
Data Sort command xx, 226
Data Table command xx, 233
Data type
code resources 253-254
converting xiv-xix, 253-256
Data type, *continued*
returning cell information 27
returning value information xvii, 240
type numbers 15, 131, 199
DATA.DELETE function xx, 43
DATA.DELETE? function 43
DATA.FIND function xx, 43
DATA.FIND.NEXT function xx, 43
DATA.FIND.PREV function xx, 43
DATA.FORM function xx, 43-44
DATA.SERIES function xx, 44
Database
adding numbers xxvi, 56, 62
averaging entries xvi, 46, 56
counting *See* Counting
criteria *See* Criteria
data forms 43-44
deleting records 43
description 55
entering records 44
extracting records xvi, 56, 72-73, 217
field 55
finding records 43, 60
functions xvi, 54-69
maximum value, returning xvi, 60
minimum value, returning xvi, 60
multiplying values xvi, 56, 61
overview 54-55
setting 55, 217
standard deviation
DSTDEV function xvi, 56, 61
DSTDEVP function xvi, 56, 62
variance
DVAR function xvi, 56, 62
DVARP function xvi, 56, 62
Database functions xvi, 54-69
Date
1900 Date System 45, 161
1904 Date System 23, 45, 161
calculating days between dates xvi
data series units 44
date functions xvi
number format 27, 85
applying 85
text value equivalent 27
Date, *continued*
returning day of week 249
serial number *See* Serial number
DATE function xvi, 44-45
DATEVALUE function xvi, 45-46
DAVERAGE function xvi, 46, 56
Day
days between two dates 46
number format
applying 85
text value equivalent 27
returning day of week xvi, 46
serial number *See* Serial number
DAY function xvi, 46
DAYS360 function xvi, 46
DBF2 file format 204
DBF3 file format 204
DBF4 file format 204
DCOUNT function xvi, 47, 56
DCOUNTA function xvi, 47, 56
DDB function xvi, 47
DDE *See* Dynamic Data Exchange (DDE)
Debugging macros xxv, 217, 229, 230
Decimal places 61, 77, 117, 251
DEFINE.NAME function xxii, 48-49, 158, 214
DEFINE.NAME? function 48
DEFINE.STYLE function xxi, 49-51, 228
Degrees, converting from radians 2, 16, 17, 18
DELETE key code 166
DELETE.ARROW function xx, 51
DELETE.BAR function xiv, 51
DELETE.COMMAND function xiv, 5, 51-52
DELETE.FORMAT function xxiii, 52
DELETE.MENU function xiv, 52-53
DELETE.NAME function xxii, 53, 219
DELETE.OVERLAY function xx, 53
DELETE.STYLE function xxi, 53-54
Deleting
arrow from chart 51
cells 63
commands xiv, 51-52, 103
custom menu bars xiv, 51



Deleting, *continued*
data series 64
database records 43
embedded objects 67
files 74
menus xiv, 52-53
names 53
overlay chart 53, 101, 102
rows 202
styles 53-54
temporary files 74
DEMOTE function xxiii, 54
Depreciation
double-declining balance method xvii, 47, 224-245
straight-line method xvii, 226
sum-of-years' digits method xvii, 232-233
DEREF function xxv, 54, 193, 217
D*function* 54-56
DGET function xvi, 56
Dialog box
adding tone 21
alert type 8-9
components 57
clearing from screen 58
creating x, 44
defining 57
displaying during macro execution x
dynamic type 58
macro interruption xiv, 8-9
user-defined, displaying xiv, xxv, 57-58, 131-132
Dialog Editor 44
DIALOG.BOX function xiv, 57-58
DIF file format 204
Direction keys *See* ARROW keys
Directory
pathname 130
returning filenames xxv, 75
saving to another 203, 204
setting path xxv, 58
DIRECTORY function xxv, 58
DISABLE.INPUT function xiv, 58-59
DISPLAY function xxii, 59-60
Display options
saving settings 205-206
specifying 59
Displaying
formula bar 117
gridlines 115
row/column headings 115
scroll bar 117
status bar 117
tool bar 117
zeroes 115
#DIV/0! error value
ATAN2 function 18
AVERAGE function 136
EXECUTE function 71
MIRR function 154
MOD function 155
overview xiii
POKE function 184
REQUEST function 198
returning value type 135
Dividing
returning remainder xviii, 155
while pasting 177
DLL *See* Dynamic Link Library (DLL)
DMAX function xvi, 56, 60
DMIN function xvi, 56, 60
Document
*See also* Worksheet
backup copy 204
calculating *See* Calculating
color palette, copying 32-33
importing styles 150-151
incorporating editions 230-231
information, returning xxv, 60-61, 107-109, 158-159
linked *See* Linking documents
names
listing all defined in document xxi, 158-159
naming document 203, 204
returning document name xxv, 60-61
new document 159
Document, *continued*
opening
existing documents 169-170
for data exchange xiv, 78-79
new documents 159
with write privileges 170
page breaks 218
page setup 173-175
printing *See* Printing
protecting *See* Protecting documents
publishing *See* Publisher
read-only access 205
saving 203, 205
scrolling *See* Scrolling
sending via Microsoft Mail 215-216
style *See* Style
types 204
viewing
different parts simultaneously xxiii, 227
multiple documents 16
workspace settings 251
write permission 204
DOCUMENTS function xxv, 60-61
DOLLAR function xix, 61
Dollar number format 61
DOS text file format 204
Double-declining balance depreciation xvi, 47, 244-245
DOWN key code 166
DPRODUCT function xvi, 56, 61
Drawing objects xxiii, 40-41, 42
Drop lines
creating 83
overlay chart 86
selecting 212
DSTDEV function xvi, 56, 61
DSTDEVP function xvi, 56, 62
DSUM function xvi, 56, 62
DUPLICATE function xxiii, 62
DVAR function xvi, 56, 62
DVARP function xvi, 56, 62
Dynamic Data Exchange (DDE)
closing a DDE channel xxv, 234
disabling input 58

Dynamic Data Exchange (DDE), *continued*
information, returning link type 110-111
opening DDE channel 70-71, 130-131
pasting document link 176
Dynamic Link Library (DLL)
calling procedures from xxiv
compatibility across operating systems 253-254
data types 253-254
registering xxv, 193-194
requesting data xxv, 197-198
running macros xxv, 164-165
sending data xxv, 183
sending keystrokes xxv, 214-215
unregistering xxv, 242


## E


ECHO function xiv, 62
Edit Clear command xx, 31
Edit Copy Chart command xx, 36
Edit Copy command xx, 36, 189
Edit Copy Picture command xxi, 37
Edit Create Publisher command xxi, 41
Edit Cut command xxi, 43
Edit Delete command xxi, 63
Edit Fill Down command xxi, 76, 189
Edit Fill Left command xxi, 76
Edit Fill Right command xxi, 76, 189
Edit Fill Up command xxi, 76
Edit Fill Workgroup command xxi, 76
Edit Insert command xxi, 133
Edit Paste command xxi, 176, 189
Edit Paste Link command xxi, 176
Edit Paste Picture command xxi
Edit Paste Picture Link command xxi
Edit Paste Special command xxi, 177, 178
Edit Repeat command xxi, 42, 64
Edit Share command 221
Edit Subscribe To command xxi
Edit Undo command xxi, 42, 241
EDIT.COLOR function xxii, 62-63
EDIT.COLOR? function 63
EDIT.DELETE function xxi, 63
EDIT.REPEAT function xxi, 64
EDIT.SERIES function xx, 64-65, 216
Edition
incorporating editions 230-231
options 65
publishing 41-42
types 65
updating 65
EDITION.OPTIONS function xxiii, 65-66
ELSE function xiv, 66, 118, 125, 126-127
ELSE.IF function xiv, 66-67, 118, 125, 126-127
EMBED function 67
Embedded chart 40, 112
Empty cell *See* Blank cell
Empty text xiii, 76
ENABLE.COMMAND function xiv, 67-68
END key code 166
END.IF function xiv, 66-67, 68, 118, 125, 126-127
ENTER key code 166
Error
handling xiv, 68-69
returning information about last error xxi, 137
value *See* Error value
ERROR function xiv, 68-69, 131, 191
Error value
*See also specific error value*
information, returning 135
ISERR function xvii
ISERROR function xvii
ISNA function xvii
overview xiii
unsupported macro function xxvi
value type 213
Error value argument type xiii
ESC key
canceling selection marquee 25
interrupting macros 25-26
key code 166
macros, single-stepping 229
EXACT function xix, 69
Excel *See* Microsoft Excel
Exclamation point (!), volatile function notation 256
EXEC function xiv, xxvi, 10, 70-71, 130
EXECUTE function xiv, xxvi, 70, 71-72, 130
Exiting Microsoft Excel 191
EXP function xviii, 72
Expanding outline xxiii, 223-224
Exponentiation xviii, 72
External reference *See* Application, external
EXTRACT function xx, 72-73
Extracting
characters from text
LEFT function xix, 137, 160
MID function xix, 77, 151-152, 160, 208
RIGHT function xix, 160, 200
database records xx, 72-73, 217


## F


F statistic 141
F1 key
accessing online Help 14
key code 166
F4 key, finding next formula 93
FACT function xviii, 73
Factorial xviii, 73
FALSE function xviii, 73, 125
FALSE value 73
FCLOSE function xiv, 73, 78
Field
counting numbers 38
description 55
multiplying selected values 61
returning largest number 60
returning smallest number 60
File
*See also* Document
access 78
backup 203, 204
changing type 203, 204
closing
FCLOSE function xiv, 73
FILE.CLOSE function 74





File, *continued*
deleting 74
determining size xiv, 98
format 204
Help *See* Help
ID number 95, 96, 97, 98, 99
name *See* Filename
opening
existing file 169-170
for data exchange xiv, 78-79
Microsoft Mail files 171-172
new file 159
protection *See* Protecting documents
reading from
FREAD function xiv, 96-97
FREADLN function xxiv-xvii, 97
saving
as template 204
without saving changes 205
workspace file 205-206
setting position xiv, 95-96
temporary 74
types 109, 159
writing to xiv, 97-98, 99, 100
File Close All command xxi, 42
File Close command xxi, 74
File Delete command xxi, 74
File Exit command xxi, 191
File ID number 78
File links *See* Linking documents
File Links command xxi, 170
File New command xxi, 159
File Object Attributes command 19
File Open command xxi, 169
File Open Mail command xxi, 171
File Page Setup command xxi, 174
File Print command xxi, 186, 218
File Print Preview command xxi, 186
File Printer Setup command xxi, 187
File Quit command xxi
File Save As command xxi, 204
File Save command xxi, 203
File Save New Object command 205
File Save Workspace command xxi, 206
File Send Mail command xxi, 215
FILE.CLOSE function xxi, 74
FILE.DELETE function xxi, 74
FILE.DELETE? function 74
Filename
designating 204-205
returning xxv, 27, 75
FILES function xxv, 58, 74, 75
Fill pattern, returning 114
FILL.DOWN function xxi, 76
FILL.LEFT function xxi, 76
FILL.RIGHT function xxi, 76
FILL.UP function xxi, 76
FILL.WORKGROUP function xxi, 76
Filling
rows/columns 76, 202
workgroup selection 76
Financial functions xvi-xvii
FIND function xix, 76-77, 209
Finding
*See also* Replacing
array values
HLOOKUP function xviii, 123
INDEX function xii, 127-129
LOOKUP function xviii, 145-146
MATCH function xviii, 148-149
VLOOKUP function xviii, 246-247
database records 43, 72
formulas
FORMULA.FIND function 92
FORMULA.FIND.NEXT function 93
FORMULA.GOTO function 93
FORMULA.REPLACE function 94
text
FIND function xix, 76-77, 209
SEARCH function xix, 76, 208-209
wildcard characters 208
FIXED function xix, 77
Folder
pathname 130
returning filenames xxv, 75
saving files 203, 204
Font
applying to selection 80-82
changing built-in fonts 196-197
colors 81
defining styles 49, 50, 78, 228
information, returning
button 113-114
default fonts 108
cell 104
FONT function 78
Footer 174
FOPEN function xiv, 78-79
FOR function xiv, 79-80
FOR–NEXT loop
ending xxv, 160
interrupting xiv, 22, 79
starting xiv, 79-80
FOR.CELL function xiv, 80
FOR.CELL–NEXT loop
ending xxv, 160
interrupting xiv, 22
starting xiv, 80
Foreground color 114, 180
Format 3-D View command xxi, 349
Format Alignment command xxi, 9
Format Border command xxi
Format Bring To Front command xxi, 22
Format Cell Protection command xxi, 28
Format Column Width command xxi, 33
Format Font command xxi, 81
Format Group command xxi, 119
Format Justify command xxi, 136
Format Legend command xxi, 82
Format Main Chart command xxi, 82, 147
Format Move command xxi
Format Number command
DELETE.FORMAT function 52
FORMAT.NUMBER function xxi, 85

Format Number command, *continued*
vs. DOLLAR function 61
vs. FIXED function 77
WEEKDAY function 249
Format Object Placement command xxi, 182
Format Object Protection command xxi, 163
Format Overlay command xxi, 64, 85, 173
Format Patterns command xxi, 179
Format Row Height command xxi, 201
Format Scale command xxi, 206, 207, 208
Format Send To Back command xxi, 216
Format Size command xxi
Format Style command
APPLY.STYLE function xxi, 14
DEFINE.STYLE function 49, 50
DELETE.STYLE function 53
MERGE.STYLES function 150
STANDARD.FONT function 228
Format Text command xxii, 88
Format Ungroup command xxii, 241
FORMAT.FONT function
changing font using macro 196
description xxi, 80-82
labeling chart items 19
vs. FONT function 78
FORMAT.LEGEND function xxi, 82
FORMAT.MAIN function xxi, 82-84
FORMAT.MOVE function xxi, 4, 84-85, 105
FORMAT.NUMBER function xxi, 50, 85
FORMAT.OVERLAY function xxi, 64, 85-86, 173
FORMAT.SIZE function xxi, 4, 87, 105
FORMAT.TEXT function xxii, 88
Formatting
*See also specific formatting*
arrows 4
border
applying 88
defining styles 49, 50
Formatting, *continued*
cell protection *See* Locking
chart 35, 82-84
clearing 31
copying format 76
information, returning
button 113-114
cell xxv, xvii, 26, 103-105
document 108-109
text box 113-114
numbers *See* Number
pasting 177
style *See* Style
text 88-90
Formula
array formula xiv, 153
cell information, returning 104-110
clearing 31
controlling calculation 23-24
displaying/hiding 59, 115
editing arguments 110
entering into cell xxiii, 89-90
filling range xxiii, 91-92
finding
FORMULA.FIND function 92
FORMULA.FIND.NEXT function 93
FORMULA.GOTO function 93
goal-seeking 118
naming
APPLY.NAMES function 13-14
DEFINE.NAME function 48
DELETE.NAME function 53
overview xiii
returning name xxvi, 111-112
nested xiv
overview xiii
pasting 177
replacing 94
testing 136
Formula Apply Names command xxii, 13
Formula bar
displaying cell contents xxvi, 110
embedded object display 67
returning workspace information 117
Formula Create Names command xxii, 39
Formula Define Name command
defining names xxii, 53
deleting names xxii, 48
deleting print area 219
returning value assigned to name 217
Formula Find command xxii, 92
FORMULA function
changing cell values 220
converting formulas 91
deleting data series 64
description xxiii, 89-90
using with hidden windows 122
Formula Goal Seek command xxii, 118
Formula Goto command xxii, 93
Formula Note command xxii
Formula Outline command xxii, 173
Formula Paste Function command xiv, 189
Formula Paste Name command xxii, 142, 189
Formula Replace command xxii, 94
Formula Select Special command xxii, 213
Formula Show Active Cell command xxii, 222
FORMULA.ARRAY function xxiii, 90
FORMULA.CONVERT function xxiii, 91
FORMULA.FILL function xxiii, 91-92
FORMULA.FIND function xxii, 92
FORMULA.FIND.NEXT function xxii, 93
FORMULA.FIND.PREV function xxii, 93
FORMULA.FIND? function 92
FORMULA.GOTO function xxii, 93, 193, 210
FORMULA.REPLACE function xxii, 94
FORMULA.REPLACE? function 94
Fortran 256
FPOS function xiv, 95-96, 99
FREAD function xiv, 95, 96-97





FREADLN function xiv, 95, 97
FREEZE.PANES function xxii, 97
FSIZE function xiv, 95, 98
FULL function xxiii, 98
Function
action-equivalent xxiii
arguments *See* Argument
array formula *See* Array formula
control macro functions xxiv-xxv
converting references to values 1
customizing functions xxiii-xxiv
data consolidation functions 35
defined ix
documentation conventions xi
financial functions xvi-xvii
logical functions xviii
lookup functions xviii
macro function *See* Macro function
mathematical functions xviii
matrix functions xviii
naming 5
reference-returning xiii
statistical functions xix
syntax xi-xii
text functions xix
trigonometric functions xx
types ix-xi
using formulas xiii
value-returning xxv-xxvi
worksheet functions *See* Worksheet functions
Future value *See* Investment
FV function xvi, 98-99
FWRITE function xiv, 95, 99
FWRITELN function xiv, 95, 99-100


## G


Gallery 3-D Area command xxii, 100
Gallery 3-D Column command xxii, 100
Gallery 3-D Line command xxii, 100
Gallery 3-D Pie command xxii, 100
Gallery Area command xxii, 101
Gallery Bar command xxii, 101
Gallery Column command xxii, 101
Gallery Combination command xxii, 35
Gallery Line command xxii, 101
Gallery Pie command xxii, 102
Gallery Preferred command xxii, 185
Gallery Set Preferred command xxii, 218
Gallery XY (Scatter) command xxii, 102
GALLERY.3D.AREA function xxii, 100
GALLERY.3D.COLUMN function xxii, 100
GALLERY.3D.LINE function xxii, 100
GALLERY.3D.PIE function xxii, 100
GALLERY.AREA function xxii, 100-101
GALLERY.BAR function xxii, 101
GALLERY.COLUMN function xxii, 101
GALLERY.LINE function xxii, 101-102
GALLERY.PIE function xxii, 102
GALLERY.SCATTER function xxii, 102
General format 185
GET.BAR function xiv, 102-103
GET.CELL function xxv, 14, 26, 103-105
GET.CHART.ITEM function xxvi, 105-106
GET.DEF function xxvi, 106-107, 111
GET.DOCUMENT function xxvi, 107-110
GET.FORMULA function xxvi, 110
GET.LINK.INFO function xxvi, 110-111
GET.NAME function xxvi, 111-112, 158
GET.NOTE function xxvi, 112
GET.OBJECT function xxvi, 112-115
GET.WINDOW function xxvi, 115-116, 241
GET.WORKSPACE function xxvi, 116-118
GOAL.SEEK function xxii, 118
GOTO function xxv, 2, 30, 66, 68, 118-119, 125
Graph *See* Chart
Graphic, copying 37
Gridlines
color 115
information, returning 115
printing 174
selecting 212
window display 59
GRIDLINES function xx, 119
GROUP function xxi, 119
GROWTH function xix, 119-120, 139, 145


## H


HALT function xxv, 26, 120-121
Header 174
Help
activating xiv, 121
Alternate Menu Key option 251
converting existing custom Help files 121
creating custom Help topics 6, 121
function directory ix
HELP function xiv, 121
Help Index command ix
HELP key code 166
Hewlett Packard NewWave
link type 142, 171, 219
macro functions supported by xxvii
saving new objects 205
saving objects 205
setting file attributes 19-20
setting file object attributes 19-20
sharing objects 221
using with Microsoft Excel 221
Hi-lo lines
main chart 83
overlay chart 86
selecting 212
Hidden cell 104
HIDE function xxii, 122
HIDE.OBJECT function xxiii, 122

Hiding
cells 28
column 34
information, returning
cell 104
object 114
window 115
names 48
objects xxiii, 122
rows 202
windows 122
HLINE function xxiii, 122
HLOOKUP function xviii, 123, 146, 246
HOME key code 166
Horizontal scrolling *See* Scrolling
Hour
converting from serial number xvi, 123-124
number format, text value equivalent 27
HOUR function xvi, 123-124
HPAGE function xxiii, 124
HSCROLL function xxiii, 2, 124-125
Hyperbolic cosine xx, 37-38
Hyperbolic sine xx, 17, 225
Hyperbolic tangent xx, 234

## I

ID number
external application 130
file 78, 95, 96, 97, 98, 99
menu bar 5, 102, 222
object 81, 114
task 70
IF function
description xviii, 125-127
structuring conditional actions 66, 67, 68
using with APP.ACTIVATE 10
using with RESTART function 198
vs. GOTO function 118
INDEX function
description xviii, 127-129
obtaining slope of line 139
INDEX function, *continued*
selecting
defined names 158
document names 60
inverse matrix elements 154
window names 250
worksheet names 142
Index number 30
INDIRECT function xvii, 129
Info Cell command 60
INFO function xvii, 129-130
Info menu bar 5
Info window
menu bar ID number 222
screen display 59, 224
window display 59-60
Information functions xvii
INITIATE function
channel number, returning 130, 197, 234
description xxv, 130-131
operating system support xxv, 130
task ID number, returning 70
INPUT function xxv, 131-132
INSERT function xxi, 133
INSERT key code 166
INT function xviii, 133, 155, 240
Integer
random xviii, 191
rounding numbers
FIXED function xix, 77
INT function xviii, 133, 240
ROUND function xviii, 200-201
signed 257
truncating number to xviii, 239-240
Interest
internal rate of return xvii, 134-135, 154
payment xvi, 133-134
rate xvii, 192
Internal rate of return xvii, 134-135, 154
Inverse hyperbolic cosine xx, 2
Inverse hyperbolic sine xx, 17
Inverse hyperbolic tangent xx, 18
Inverse matrix 153-154
Investment
cash flow 162-163
future value
annuity 192
calculating payments 183
FV function xvi, 98-99
interest *See* Interest
internal rate of return xvii, 134-135, 154
net present value xvii, 162-163
payments *See* Payment
present value
defined 98, 183, 192
returning xvii, 189-190
IPMT function xvi, 133-134, 190
IRR function xvi, 134-135, 162
IS functions 135-136
ISBLANK function xvii, 135
ISERR function xvii, 135
ISERROR function xvii, 135
ISLOGICAL function xvii, 135
ISNA function xvii, 135
ISNONTEXT function xvii, 135
ISNUMBER function xvii, 135, 136
ISREF function xvii, 135
ISTEXT function xvii, 135
Italic formatting
formatting font 81, 230
returning cell information 104
returning default font information 108
Iteration 23, 79, 108

## J

JUSTIFY function xxi, 136

## K

Keyboard
key codes 165-166
macro key combinations xxv, 48, 165-167





## L

Label
chart items 18, 19
creating names from 39
formatting 181
moving 85
selecting 211
sizing 88
LAST.ERROR function xxvi, 137
Least squares procedure 138
LEFT function xix, 137, 160
LEFT key code 166
Left margin 174
LEGEND function xx, 137
Legend
adding to chart 137
positioning 82
selecting 211
LEN function xix, 137
Line
calculating 138-141
drawing 40
formatting 180-181
object type 112
slope 139
straight line 238-239
Line chart
chart type 82, 85
creating 101-102
drop lines 86
formatting 83
hi-lo lines 86
Linear regression xix, 140-141
LINEST function xix, 138-141, 144
Linking documents
changing links 28-29
consolidating data 15
information, returning xxvi, 110-111, 142
opening documents 170-171
outside applications *See* Application, external
pasting links 176-177
recalculation 109
types of links 28, 142, 170, 219, 242
updating xxiii, 169, 219, 220, 242
LINKS function xxi, 142, 170, 171
List box
custom dialog boxes 57
macro function argument equivalent x
LIST.NAMES function xxii, 142-143, 158
LN function xviii, 72, 143
Locking
*See also* Protecting documents
cells
defining styles 49
information, returning 26, 104
moving to unlocked cell xxiii, 241
procedure 28
objects 112, 163, 188
LOG function xviii, 143
LOG10 function xviii, 144
Logarithm
base-10 xviii, 144
natural xviii, 72, 143
specified base xviii, 143
LOGEST function xix, 139, 144-145
Logical functions xviii
Logical value
argument syntax xii
data type xiii, xiv-xv, 240
naming 48
reference xiii
returning
AND function xviii, 9-10
FALSE function xviii, 73
IF function xviii, 125-126, 127
NOT function xviii, 160
OR function xviii, 172
TRUE function xviii, 239
value information xvii, 135, 240
selecting 211
Looking up values
HLOOKUP function xviii, 123, 146, 246
LOOKUP function xviii, 145-147
MATCH function xviii, 148
VLOOKUP function xviii, 123, 147, 246-247
LOOKUP function xviii, 145-147
Lookup functions xviii
Looping macros
counters 79
creating x
ending xxv, 160
FOR–NEXT loop xiv, 79-80
FOR.CELL–NEXT loop xiv, 80
logical tests xxv, 66, 67
specifying action on interruption xxiv
starting xxiv
terminating xiv, 22
LOWER function xix, 147
Lowercase characters
converting text xix, 147
initial capitals 188
searching text xix, 209

## M

Macintosh
character set 29, 32
data types 255-256
system software version 7.0 functions xxvi-xxvii
text file format 204
Macro
adding tone xxiii, 21
assigning to object 17, 41
Auto_Close 120, 191
Auto_Open 120
branching macros 126
calling other macros 198, 200
debugging xxv, 136, 217, 229-230
dialog box *See* Dialog box
ending current macro xxv, 200
examples
appending text to end of file 95
Auto Open with reference style query 9
converting references 91
disabling keyboard macro interruption 26
displaying character code numbers in file 96-97
extracting database records 72
FOR–NEXT loop 79
freezing row/column titles on screen 97

Macro, *continued*
examples, *continued*
hiding all displayed documents 60
labeling chart items 19
listing files in directory/folder 75
opening channel to Microsoft Word 198
preventing macro interruption 69
turning off error checking 69
undoing multiplication 42
unhiding all windows 79
updating edition 66
functions *See* Macro function
halting xxv, 120-121
interrupting
ALERT function xiv, 8, 9
designating next action 25-26
preventing interruption xiv, 25-26
key combinations xxv, 165-167
looping *See* Looping macros
pausing xxv, 248
recording
Absolute Record option 202
custom commands xiv, 42
mouse movements 34, 202
moving objects 84
named formulas 48
Relative Record option 202
RUN function 203
selecting cells 210
sizing objects 87
running
at specific intervals xxv, 168
at specific time xxv, 168
checking values 198
custom commands 6
error handling xiv, 68-69
from macro sheet 203
halting 198
on interruption of other macro 25, 26
on recalculation xxv, 167
on receiving new remote data xxv, 164-165

Macro, *continued*
running, *continued*
on window activation xxv, 168-169
redirecting xxv, 118-119
restarting after RETURN statement xxv, 198
RUN function procedure 202-203
screen updating 62
shortcut keys 48
single-step mode xxv, 68, 203, 229-230
speeding up xiv, 62, 122
with mouse 17
stopping 200
structuring multiple actions xiv, 66
subroutine, ending 200
temporary files 74
types 48
writing macros 127
Macro Absolute Record command 210
Macro Assign To Object command xxii, 17
Macro function
action-equivalent x, xxiii
arguments x
command-equivalent functions x, xx-xxii
control functions x, xxiv-xxv
customizing functions x, xxiii-xxiv
description ix
dialog-box functions x
operating system differences xxv-xxvii
overview ix
redirecting x
types x-xi
value-returning x, xi, xxv-xxvi
Macro Relative Record command 210
Macro Run command xxii, 200, 202, 203, 229
Macro sheet
changing cell values 220-221
clearing 31
creating xxi, 159
custom command definition 5

Macro sheet, *continued*
custom menu definition 6
defining criteria 217
defining database 217
errors *See* Error
gridlines 174
information, returning
GET.BAR function 103
GET.DEF function 107
GET.DOCUMENT function 107-110
GET.FORMULA function 110
GET.NAME function 111-112
menu bar 5, 222
naming values xxv, 217-218, 220
opening new window 159-160
pasting items 177-178
protecting 188, 189
returning array 217
selecting
cells xxiii, 209-210, 212-214
objects xxiii, 211-212
storing values 220-221
Macro sheet menu bar 5, 222
MAIN.CHART function 147
MAIN.CHART.TYPE function 147
Manual calculation *See* Calculating
Margin 174
Marquee, canceling xxiii, 25
MATCH function xviii, 148
Math coprocessor 117
Mathematical functions xviii
Matrix
determinant xviii, 149-150
functions xii
inverse xviii, 153-154
multiplying xviii, 155, 232
transpose of array
description xviii, 237-238
returning 60, 75, 158, 250
Matrix functions xviii
MAX function xiii, xix, 35, 149
Maximizing
external application windows 70
Microsoft Excel window 11
returning window information 116, 117

MDETERM function xviii, 149-150
Mean *See* Averaging
MEDIAN function xix, 150
Memory
allocation 70
amount available 130
amount in use 130
freeing memory xxv, 242
total amount 130
Menu
adding commands xxiii, 5-6, 103
adding to menu bar xxiii, 6-7
alternate menu key 117
bar *See* Menu bar
checked commands xiv, 29
command position 67
custom menus
adding to menu bar xxiii, 6-7
creating x
custom Help topics 6
position 6
deleting xiv, 52-53
dimmed commands 67
enabling/disabling commands xiv, 67-68
full menus 222
position 6, 7, 67
renaming xiv, 195
restoring deleted commands 5
restoring deleted menus 4, 6-7
short menus 222
using name in functions 5
Menu bar
adding menus xxiii, 6-7
creating xxiii, 4-5, 222
deleting xiv, 51
displaying xiv, 4, 222-223
ID number 4, 5, 102, 222
maximum number definable 5
menus
adding xxiii, 6-7
deleting xiv, 52-53
restoring 4, 6-7
returning bar number xiv, 102-103
types 5
Menu key 251

MERGE.STYLES function xxi, 150-151
MESSAGE function xiv, 8, 151
Messages, alert xiv, 151
Microsoft Excel version 2.x file format 204
Microsoft Excel
arguments to called procedures 256
blocking input xiv, 58-59
link type 28, 171
Microsoft Excel for OS/2
code resources 193
data types 253-254
date system 161
defining colors 63
unsupported macro functions xxvi-xxvii
Microsoft Excel for the Macintosh
code resources 193
data types 255-256
date system 161
defining colors 63
sending documents via Microsoft Mail 215-216
system software version xxvi, xxvii
unsupported macro functions xxvi-xxvii
version 1.5 230
Microsoft Excel for Windows
code resources 193
data types 253-254
date system 161
defining colors 63
unsupported macro functions xxvi-xxvii
Microsoft Excel Solver 118
NewWave *See* Hewlett Packard NewWave
returning version 117, 130
version number 117
Microsoft Excel Custom Help Development kit 121
Microsoft Mail
opening Mail files 171-172
sending documents 215-216

Microsoft Windows
character set 29, 32
data types 253-254
text file format 204
unsupported functions xxvi-xxvii
MID function xix, 77, 151-152, 160, 208
MIN function xiii, xix, 35, 152
Minimizing
external application windows 70
Microsoft Excel window 11
returning window information 116, 117
Minute
converting from serial number xvi, 152-153
number format, text value equivalent 27
MINUTE function xvi, 152-153
MINVERSE function xviii, 153-154
MIRR function xvi, 154
MMULT function xviii, 155
MOD function xviii, 155
Modes 117
Modified internal rate of return 154
Modulus 155
Month
converting from serial number xvi, 156
number format, text value equivalent 27
MONTH function xvi, 156
Mouse
blocking input from xiv, 58-59
moving objects 84
recording macros 34
returning workspace information 117
running macros 17
setting column width 34
sizing objects 87
splitting windows xxiii, 227
MOVE function xx, 156-157, 159
MOVE? function 156
Moving
chart objects 84-85
defining selection 43

Moving, *continued*
Microsoft Excel window 11-12
notes xxvi, 112
object placement 182-183
returning object information 113
windows 156-157
worksheet objects 84
Multiplying
DPRODUCT function xvi, 61
PRODUCT function xviii, 35, 187
SUMPRODUCT function xviii, 232
undoing 42
while pasting 177


## N


N function xvii, 157
NA function xvii, 157, 158
#N/A error value
ACTIVE.CELL function 3
APP.MAXIMIZE function 11
APP.MINIMIZE function 11
ARGUMENT function 15
description 157-158
EXECUTE function 71
FILES function 75
FOPEN function 78
FREAD function 96
FREADLN function 97
FWRITE function 99
GET.BAR function 103
GET.DOCUMENT 109
GET.LINK.INFO function 110
GET.OBJECT function 113, 114, 117
GET.WORKSPACE function 117
GOTO function 118
HLOOKUP function 123
INITIATE function 130
LAST.ERROR function 137
LINKS function 142
LOOKUP function 146
MATCH function 148
overview xiii
REQUEST function 198
returning value type xvii, 135, 157-158
VLOOKUP function 246
#NAME? error value
DELETE.STYLE function 53
overview xiii
returning value type 135
NAMES function xxvi, 158-159
Naming
APPLY.NAMES function 13-14
array 217
cell values 217
creating names from labels 39
current selection 48
defining names
DEFINE.NAME function 48-49
returning definition xxvi, 111-112
SET.NAME function xxv, 217-218
deleting names 53
hidden names 48, 158
information, returning 107
listing xxvi, 142-143, 158-159
logical values 48
naming conventions 48
numbers 48
object 113
using names in arguments xiii, 55
values 217
Natural logarithm xviii, 72
Net present value xvii, 162-163
NEW function xxi, 159
NEW.WINDOW function xxii, 159-160
NewWave *See* Hewlett Packard NewWave
NEXT function xxv, 160
Nonblank cell, counting *See* Counting
Nonprintable characters, removing from text 30
Normal file format 204
Normal style, setting attributes 228
NOT function xviii, 160
Note
clearing 31
creating 160-161
moving contents xxvi, 112
pasting 177
NOTE function xxii, 160-161
NOTE? function 160
NOW function xvi, 161, 248
NPER function xvii, 161-162, 190
NPV function xvii, 134, 162-163
#NULL! error value
overview xiii
returning value type 135
Null menu bar 5
#NUM! error value
DGET function 56, 57
GROWTH function 119
IRR function 134
MINVERSE function 153
overview xiii
RATE function 192
returning value type 135
SQRT function 228
NUM LOCK key code 166
Number
absolute value, returning xviii, 1
adding
DSUM function xvi, 62
SUM function xviii, 231-232
SUMPRODUCT function xviii, 232
while pasting 177
argument type xii, xiii
average
AVERAGE function xix, 20, 35, 193
DAVERAGE function procedure xvi, 46, 56
data type xiv-xv
conversion values
from text xix, 243
from value xvii, 157
to text xix, 234-235
counting
COUNT function xix, 38
COUNTA function xix, 38-39
DCOUNT function xvi, 47
DCOUNTA function xvi, 47
decimal places xix, 77
dividing
returning remainder xviii, 155
while pasting 177
entering into cell xxiii, 89-90





Number, *continued*
exponentiation xviii, 72
factorial xviii, 73
formatting
applying 85
defining styles 49, 50
deleting custom format xxiii, 52
DOLLAR function xix, 61
Format Number command 61, 77
General format 185
returning cell information 26, 27, 104
TEXT function xix
median xix, 150
minimum, returning 152
multiplying
DPRODUCT function xvi, 61
PRODUCT function xviii, 35, 187
SUMPRODUCT function xviii, 232
undoing multiplication 42
while pasting 177
naming 48
random xviii, 191
rounding
FIXED function xix, 77
INT function xviii, 133, 240
ROUND function xviii, 200-201
sign 225
square root xviii, 228
subtracting 177
text conversion 243
truncating xviii, 239-240


## O


Object
assigning macro 41
Autoshared option 19
background *See* Background
border *See* Border
creating xxiii, 40-41
duplicating xxiii, 62
Object, *continued*
embedded objects 67
file object attributes 19-20
grouping 119
ID number 114
identifier, returning as external reference xxvi, 214
information, returning xxvi, 112-115
locking 112
moving 113, 182-183
name 113
offset 113
placement 216
rearranging 22
saving 205
sizing 113, 182-183
types 40, 112, 214
Object ID number 81
OBJECT.PROTECTION function xxi, 163
OFFSET function xvii, 1, 163-164, 210
OK button 8, 57, 131
ON.DATA function xxv, xxvi, 164-165
ON.KEY function xxv, 165-167
ON.RECALC function xxv, 167
ON.TIME function xxv, 168
ON.WINDOW function xxv, 168-169
OPEN function xxi, 58, 78, 169-170
OPEN.LINKS function xxi, 142, 170-171
OPEN.MAIL function xxi, xxvi, 171-172
Opening
files
existing files 169-170
for data exchange xiv, 78-79
Microsoft Mail files 171-172
new files 159
with write privileges 170
linked documents 170-171
templates 159
windows 159-160
Oper data structures 257
Operating system
code resource compatibility 253-254
color conversion 63
information, returning xvii, 117, 130
macro function compatibility xxvi-xxvii
type 130
version number 130
Option button
dialog-box definition 57
macro function arguments x
OPTION key combinations, assigning to macros 165
Options Calculate Document command xxii, 23
Options Calculate Now command xxii, 23, 191
Options Calculation command xxii, 23, 185
Options Color Palette command xxii, 32-33, 63
Options Display command xxii, 59
Options Font command 78
Options Freeze Panes command xxii, 97
Options Full Menus command xxii, 222
Options Protect Document command xxii, 28, 188, 189
Options Remove Page Break command xxii, 195
Options Set Page Break command xxii, 195, 218
Options Set Print Area command xxii, 218
Options Set Print Titles command xxii, 219
Options Short Menus command 222
Options Standard Font command xxii, 228
Options Unfreeze Panes command 97
Options Unprotect Document command 188, 189
Options Workspace command xxii, 251
OR function xviii, 172

OS/2 operating system
  character set 29, 32
  data types 253-254
  text file format 204
  unsupported functions xxvi-xxvii
Outline
  collapsing xxiii, 223-224
  controlling display xxiii, 224-225
  creating 173
  demoting row/column xxiii, 54
  displaying 109
  expanding xxiii, 223-224
  promoting row/column xxiii, 187, 188
  symbols 59, 116
OUTLINE function xxii, 173
Outline format
  formatting font 81
  returning cell information 104
  returning default font information 108
Oval
  drawing 40
  information, returning 112
  moving 84
Overlapped chart 86
Overlay chart
  adding data series 64
  adding overlay 7
  attaching text 18
  deleting 53, 101, 102
  formatting 85-86
  information, returning 108
  series distribution 86
  types 85
  view 85, 86
OVERLAY function 173
OVERLAY.CHART.TYPE function 173

## P

Page break
  automatic 59
  overriding 218
  previewing before printing 186
  removing 195
  setting 218
PAGE DOWN key code 166
PAGE UP key code 166
PAGE.SETUP function xxi, 173-175
Pane
  activating 2-3
  freezing 97
  information, returning 116
Paper feed 186
Paper size 174-175
Parentheses (( ))
  enclosing multiple references xiii, 14
  number format 26, 27
  with union operators xv
PARSE function xx, 175-176
Parsing data 175-176
Pascal 256
Password
  entering 170
  PROTECT.DOCUMENT function 189
  SAVE.AS function 203, 204
  write permission 204
PASTE function xxi, 36, 43, 176
PASTE.LINK function xxi, 176-177
PASTE.PICTURE function xxi, 177
PASTE.PICTURE.LINK function xxi, 177
PASTE.SPECIAL function
  defining selection 36
  description 176, 177-179
  transposing array 237
Pasting
  cells 133
  chart items 179
  defining selection 43
  file links 176-177
  selection 176
  worksheet items 177-178
Path
  setting xxv, 58
  name 107, 130
Pattern
  cell 104
  defining styles 49, 50
  fill 114
  type 180
PATTERNS function xxi, 4, 179-182
PATTERNS? function 179
Pausing macro xxv, 248
Payment
  computing on 360-day year 46
  expressed as negative numbers 134, 190
  returning
    interest payment xvi, 133-134
    number of payments xvii, 161-162
    periodic total payment xvii, 183
    principal payment xvii, 184-185
Percent sign (%) key code 166
Percentage number format 27
PI function xviii, 182
Pi, returning xviii, 182
Picture chart
  chart type 181
  copying 37
  creating 40
  picture units 181
  stacking/stretching in chart 176
Pie chart
  chart type 82, 85, 102
  creating 102
  first slice angle 86
  formatting 83
  moving slice 85
  view 83, 86
Placeholder 59, 109
PLACEMENT function xxi, 182-183
Plot area 18, 181
Plus sign (+) key code 166
PMT function xvii, 183, 190
Point 84, 87
Pointer 257
POKE function xxv, xxvi, 183-184
Population
  standard deviation
    DSTDEV function xvi, 56, 61
    DSTDEVP function xvi, 56, 62
    STDEV function xix, 228
    STDEVP function xix, 229





Population, *continued*
  variance
    DVAR function xvi, 62
    DVARP function xvi, 62
    VAR function xix, 243
    VARP function xix, 244
Powers *See* Exponentiation
PPMT function xvii, 184-185, 190
Precision As Displayed 23, 108, 185
PRECISION function xxii, 185
PREFERRED function xxii, 185
Present value *See* Investment
Principal, returning payment *See* Payment
PRINT function xxi, 186, 218
PRINT SCREEN key code 166
PRINT.PREVIEW function xxi, 186
PRINTER.SETUP function xxi, xxvi, 187
Printing
  active document 186
  background process 248
  color printing 186
  entire worksheet 219
  multiple copies 186
  multiple selections 219
  nonprintable characters *See* Nonprintable characters
  page range 186
  page setup 173-175
  paper feed 186
  print area, setting 218-219
  print preview 186
  print titles, defining 219
  quality 186
  selecting printer 187
Probability 225
Product *See* Multiplying
PRODUCT function xviii, 35, 187
Program *See* Application, external
Promote button 187
PROMOTE function xxiii, 187-188
PROPER function xix, 188
PROTECT.DOCUMENT function xxii, 28, 163, 188-189
Protecting cells *See* Locking; Hiding
Protecting documents 28, 163, 188-189
Publisher
  canceling 65, 66
  creating 41-42
  link type 142, 171
  opening 66
  setting edition options xxiii, 65-66
PV function xvii, 162, 189-190

## Q

Question mark (?)
  dialog-box function notation x
  wildcard character
    deleting files 74, 75
    opening files 169
    replacing text 94
    returning filenames 75
    searching text 208
QUIT function xxi, 191
Quitting Microsoft Excel 191
Quotation marks
  double (")
    enclosing double quotation marks xii-xiii
    enclosing field headings 55
    pathname notation 58
    right-aligned text 27
    text argument notation xii-xiii
  empty ("")
    empty text notation xiii, 76
    right-aligned text 27
  single ('), left-aligned text 27

## R

R-squared statistic 141
R1C1 reference style
  cell address 7
  converting text to xxvi, 235-236
  converting to A1 style xxiii, 91
  information, returning 117
  recording macros 203
  setting row height with mouse 202
  specifying xxiii, 1
Radians, converting to degrees 2, 16, 17, 37, 234
Radio button *See* Option button
RAND function xviii, 191
Random number xviii, 191
Range
  area, defined 14
  border *See* Border
  criteria 55
  filling in xxiii, 91-92
  index number 30
  information, returning
    column number 33
    number of columns 34
  justifying text 136
  name, returning xxvi, 111-112
  selecting 35-36
  values, selecting 30
RATE function xvii, 190, 192
Read-only file access 109, 169
Recalculation *See* Calculating
Record *See* Database
Rectangle
  drawing 40
  information, returning 112
#REF! error value
  CALLER function 25
  EXECUTE function 71
  GET.OBJECT function 114
  INDEX function 128, 129
  INDIRECT function 129
  OFFSET function 164
  overview xiii
  POKE function 184
  REQUEST function 198
  returning value type 135
  VLOOKUP function 246
Reference
  absolute *See* Absolute reference
  active cell, returning xxv, 3-4
  argument type xiii
  calling function, returning reference xxv, 24-25
  consolidation 109
  converting
    from text xxvi, 1, 192-193, 214, 235-236
    from value xxvi, 1, 193, 214, 235-236

Reference, *continued*
converting, *continued*
to text xxvi,
to value xxv, 54, 193, 217
defined xiii, 105
entering into cell xxiii, 89-90
external
active cell 3
defining names 48
quoted external references 35
returning worksheet names 142
information, returning
active cell xxv, 3-4
active menu bar xiv, 102-103
areas, number xvii, 14-15
column number xvii, 33
columns, number of xvii, 34
contents 104
filename 27
multiple selection 14
object reference 112
position 103, 104
remote reference updating 108
row number xvii
rows, number of xvii, 202
type 104, 135-136
looking up values
INDEX function xviii, 127-128
LOOKUP function xviii,
MATCH function xviii,
multiple selection xiii, 14
naming xiii, 5, 13-14
relative *See* Relative reference
returning
ACTIVE.CELL xxv, 3
CALLING function xxv, 24-25
INDIRECT function xvii, 129
OFFSET function xvii, 1, 163-164, 210
overview xiii
SELECTION function xxvi, 214
TEXTREF function xxvi, 1, 192, 214, 235-236
REFTEXT function xxvi, 1, 3, 192-193, 214
REGISTER function
data types
Microsoft Excel for the Macintosh 255-256
Microsoft Excel for Windows 253-254
OS/2 253-254
description xxv, 193-194
UNREGISTER function 242
using with CALL function 24,
Regression
curve xix, 119, 120, 144-145
linear xix, 138-141
Relative reference
absolute reference, converting to
ABSREF function 1
automatic conversion 54
DEREF function xxv, 54
FORMULA.CONVERT function xxiii, 91
REFTEXT function xxvi, 1, 3, 192-193, 214
cell address 7
column width, recording 33
converting to
FORMULA.CONVERT function xxiii, 91
RELREF function 194-195
defining names 217
returning 1, 163
row height, recording as 201, 202
RELREF function xxvi, 194-195
Remainder 155
Remote data xxv, 164-165
Remote DDE requests 117
Remote reference 23, 108
REMOVE.PAGE.BREAK function xxii, 195
RENAME.COMMAND function xiv, 195
Repeating
commands
CUSTOM.REPEAT function xiv, 42
EDIT.REPEAT function 64
text xix, 197
REPLACE function xix, 196, 208, 209
REPLACE.FONT function 196, 197
Replacing
FORMULA.REPLACE function xxii, 94
REPLACE function xix, 196
SEARCH function 208, 209
SUBSTITUTE function xix, 231
REPT function xix, 197
REQUEST function xxv, xxvi, 197-198
Reserved file format 204
Residual sum of squares 139
RESTART function xxv, 198
RESULT function xxv, 198-199
RETURN function
description xxv, 200
reactivating error checking 68
reactivating macro interruption 26
removing RETURN statements from stack 198
vs. HALT function 120
Reversing arguments 160
RIGHT function xix, 160, 200
RIGHT key code 166
Right margin 174
ROUND function xviii, 200-201
Rounding numbers
FIXED function xix, 77
INT function xviii, 133, 240
ROUND function xviii, 200-201
Row
counting rows xvii, 116, 202
deleting 63
filling 76
freezing titles 97
headings
displaying 59, 115
printing 174
reference style xxiii, 1
height, setting 201-202
hiding 202
information, returning
first used row 108
height 104





Row, *continued*
information, returning, *continued*
last used row 108
number xvii, 26, 201
outline
demoting xxiii, 54
displaying levels 224
expanding/collapsing 223
promoting x, xiii, 187-188
scrolling window by row xxiii, 246
separator 140, 145
sorting 226-227
unhiding 202
ROW function xvii, 201
ROW.HEIGHT function xxi, 201-202
ROWS function xvii, 158, 202, 250
RUN function xxii, 202-203

## S

Salvage value 47
SAVE function xxi, 203
SAVE.AS function xxi, 58, 203, 204-205
SAVE.NEW.OBJECT function xxi, 205
SAVE.WORKSPACE function xxi, 205, 206
Saving
active document 203
before closing 31-32
changes 191
directory 58
edition 65
new filename 204-205
objects xxi, 205
recalculating before saving 24
workspace xxi, 205, 206
SCALE function xxi, 206-208
Scaling
axes 206-208
page setup 175
Scatter chart 102
Screen
attributes
updating xiv, 62
Scroll bar 117
SCROLL LOCK key code 166
Scrolling
by column number xxiii, 124-125
by percentage xxiii, 124-125
freezing row/column titles 97
horizontally xxiii, 124
macro function equivalent x
scroll bar 117
selecting reference 93
vertically xxiii, 246, 247-248
window xxiii, 122
SEARCH function xix, 76, 208-209
Searching
array values
HLOOKUP function xviii, 123
LOOKUP function xviii, 145-146
MATCH function xviii, 148-149
VLOOKUP function xviii, 246-247
database records 43, 72
formulas
FORMULA.FIND function 92
FORMULA.FIND.NEXT function 93
FORMULA.GOTO function 93
FORMULA.REPLACE function 94
text
FIND function xix, 76-77, 209
SEARCH function xix, 76, 208-209
wildcard characters 208
Second
converting from serial number xvi, 209
number format, text value equivalent 27
SECOND function xvi, 209
SELECT function
consolidating data 36
deleting data series 64
description xxiii, 209-212
selecting arrow 51
using with SHOW.INFO function 224
SELECT function, *continued*
vs. FORMULA.GOTO function 93, 193
vs. SELECT.PLOT.AREA function 213
SELECT.CHART function xx, 212
SELECT.END function xxiii, 212-213
SELECT.LAST.CELL function xxii, 213
SELECT.PLOT.AREA function xx, 213
SELECT.SPECIAL function xxii, 213
Selecting
arrow 51
cells xxiii, 209-210, 212-214
chart items 51, 211-212, 213
charts 212
database records 43
FORMULA.GOTO function 93
unlocked cell xxiii, 241
values xviii, 30
SELECTION function xxvi, 214
SEND.KEYS function
DDE function 70, 71
description xxv, 214-215
making dialog-box selections 72
SEND.MAIL function xxi, xxvi, 215-216
SEND.TO.BACK function xxi, 216
Serial number
conversion to
date xvi, 44-45
day of month xvi, 46
day of week xvi, 249
hour xvi, 123-124
minute xvi, 152-153
month xvi, 156
second xvi, 209
time xvi, 236-237
year xvi, 251-252
returning
current date xvi, 161
date xvi, 45-46
time xvi, 237
today's date xvi, 237
Series *See* Data series

Series axis
  attaching text 19
  scaling 207-208
SERIES function 64, 216-217
SET.CRITERIA function xx, 217
SET.DATABASE function xx, 217
SET.EXTRACT function xx, 217
SET.NAME function xxv, 54, 214, 217-218, 220
SET.PAGE.BREAK function xxii, 195, 218
SET.PREFERRED function xxii, 185, 218
SET.PRINT.AREA function xxii, 218-219
SET.PRINT.TITLES function xxii, 219
SET.UPDATE.STATUS function xxiii, 219-220
SET.VALUE function xxv, 220-221
Shadow
  border 114
  formatting font 81
  pattern 181
  returning cell information 104
  returning default font information 108
SHARE function xxvii, 221
SHARE.NAME function xxvii
SHIFT key combinations
  assigning to macros 165-166
  CTRL+SHIFT+ENTER, entering arrays 129, 69
  CTRL+SHIFT+F7, finding next formula 92
  SHIFT+F5, finding formula 92
  SHIFT+F7, finding previous formula 93
  SHIFT+TAB, moving to previous unlocked cell 241
Short menus xxii, 222
SHORT.MENUS function xxii, 222
Shortcut keys 6
SHOW.ACTIVE.CELL function xxii, 222
SHOW.BAR function xiv, 4, 222-223
SHOW.CLIPBOARD function xxii, 157, 223
SHOW.DETAIL function xxiii, 223-224
SHOW.INFO function xxii, 224
SHOW.LEVELS function xxiii, 224-225
SIGN function xviii, 225
Sign, returning for number xviii
Signed integer 257
SIN function xx, 225
Sine xx, 16, 17, 225
SINH function xx, 17, 225
SIZE function xx, 159, 225-226
Sizing
  arrows 4, 88
  chart items 4, 87-88
  windows
    APP.MAXIMIZE function 11
    APP.RESTORE function 12
    APP.SIZE function 12-13
    ARRANGE.ALL function 16
    FULL function xxiii, 98
    SIZE function 225-226
    title bar 98
  worksheet objects
    absolutely 87-88
    automatically 114
    information, returning 113, 114
    placement 182-183
    relatively 87
SLN function xvii, 226
Slope of line 139, 140
SORT function xx, 226-227
Sorting 226-227
Space, removing from text xix, 239
SPLIT function xxiii, 97, 227
Splitting windows xxiii, 97, 116, 227
SQRT function xviii, 228
Square root xviii, 228
Standard deviation
  DSTDEV function xvi, 56, 61
  DSTDEVP function xvi, 56, 62
  STDEV function xix, 35, 228, 229
  STDEVP function xix, 35, 228, 229
STANDARD.FONT function xxii, 228
Statistical functions xix
Status bar xviii, 6, 117, 151
  displaying 117
  displaying messages xiv, 151
  text for custom menus 6
STDEV function xiii, xix, 35, 228, 229
STDEVP function xiii, xix, 35, 228, 229
STEP function xxv, 229-230
Straight-line depreciation xvii, 226
Strikethrough format
  formatting font 81
  returning cell information 104
  returning default font information 108
Style
  applying 14
  attributes 50
  creating 50-51
  defining based on active cell 49-50
  deleting 53-54
  importing styles 150-151
  outline 173
STYLE function 230
Subroutine
  defining arguments xviii, 15-16
  specifying data type returned xxv, 198-199
SUBSCRIBE.TO function xxi, 65, 230-231
Subscriber
  canceling 65, 66
  link type 142, 171
  setting edition options xxiii, 65-66
SUBSTITUTE function xix, 231
Subtracting 177
SUM function xiii, xiv, xviii, 35, 231-232
Sum-of-years' digits depreciation xvii, 232-233
SUMPRODUCT function xviii, 232
SYD function xvii, 232-233
SYLK file format 204
System menu *See* Control menu





## T

T function xvii, 233
T statistics 141
TAB key
  key code 166
  moving to next unlocked cell 241
Table
  creating 233-234
  dialog-box definition table 57
  excepting from automatic calculation 23
TABLE function xx, 233-234
TAN function xx, 234
Tangent xx, 17, 18, 234
TANH function xiv, 234
Task ID number 70, 130
Template
  creating xxi, 159
  file format 204
  opening 159
  saving file as template 204
Temporary file 74
TERMINATE function xxv, xxvi, 234
Text
  alignment *See* Alignment
  appending to file 95
  argument notation xii-xiii
  box *See* Text box
  character codes xix, 32, 96-97
  converting from number xix, 234-235
  converting from reference
    ADDRESS function xvii, 7-8
    REFTEXT function xxvi, 1, 3, 192-193, 214
  converting to number xix, 243
  converting to reference xxvi, 192, 214, 235-236
  empty text xiii, 76
  entering into cell xxiii, 89-90
  extracting characters
    LEFT function xix, 137, 160
    MID function xix, 77, 151-152, 160, 208
    RIGHT function xix, 160, 200
Text, *continued*
  finding *See* Finding
  formatting 88-90
  information, returning 104, 108-109, 113
  justifying 136
  lowercase *See* Lowercase characters
  orientation 88, 113
  reading from file xiv, 96-97
  removing nonprintable characters xix, 30
  removing spaces xix, 239
  repeating xix, 197
  replacing text *See* Replacing
  searching *See* Searching
  separator 169
  string
    comparing strings xix, 69
    last character, returning xix, 200
    locating character xix, 208-209
    returning length xix, 137
  text functions xix
  uppercase *See* Uppercase characters
  value
    information, returning xvii, 240
    selecting 213
    type 213
  wrapping in cells 9
  writing to file xiv, 99-100
Text box
  creating 40
  font 81
  formatting 81-82, 88, 181
  information, returning 112
  locking 163
  macro function arguments x
  updating xviii, 235
Text file format 204
TEXT function xix, 234-235
TEXT.BOX function xxiii, 235
TEXTREF function xxvi, 1, 192, 214, 235-236
Tickmark label 89
Tilde (~)
  key code 166
  replacing characters 94
Time
  number format
    applying 85
    text value equivalent 27
  running macros xxv, 168
  serial number *See* Serial number
  time functions xvi
TIME function xvi, 236-237
TIMEVALUE function xvi, 237
Title bar 98
TODAY function xvi, 237
Today's date xvi
Tone 21
Tool bar x, 117
Top margin 174
Transparent background 81
TRANSPOSE function
  description xviii, 237-238
  transposing arrays 60, 75, 158, 250
TREND function xix, 139, 141, 238-239
Trigonometric functions xx
TRIM function xix, 239
TRUE function xviii, 125, 239
TRUE logical value, returning
  AND function xviii, 9-10
  OR function xviii, 172
  TRUE function xviii, 125, 239
TRUNC function xviii, 239-240
Truncating numbers xviii
TYPE function xvii, 240

## U

Underline
  formatting font 81
  returning cell information 104
  returning default font information 108
UNDO function xv, 241
Undoing actions
  CUSTOM.UNDO function xiv, 42-43
  UNDO function xxi, 241
UNGROUP function xxii, 241

UNHIDE function xxii, 241
UNLOCKED.NEXT function xxiii, 241
UNLOCKED.PREV function xxiii, 241
UNREGISTER function xxv, 193, 242
UP key code 166
UPDATE.LINK function xxi, 242
UPPER function xix, 188, 242-243
Uppercase characters
  converting text xix, 242-243
  initial capitals xix, 188
  searching text xix, 209
Useful life 47

## V

Value
  absolute xviii, 1
  assigning to reference xxv, 217, 220-221
  constant value 217
  converting
    from reference xxv, 54
    to number xvii, 157
    to reference xviii
    to text xix, 233, 234-235
  counting *See* Counting
  data series xxv, 44, 220-221
  data types *See* Data type
  database *See* Database
  defining name xxv, 217-218
  entering into cell xxv, 44,
  error *See* Error value
  future *See* Future value
  index number 30
  information, returning xxvi, 111-112
  logical *See* Logical value
  matching
    HLOOKUP function xviii, 123
    LOOKUP function xviii, 145-146
    MATCH function xviii, 148-149
    VLOOKUP function xviii, 246-247
Value, *continued*
  naming *See* Naming
  negative 162
  nontext 135
  not available *See* #N/A error value
  optimal 118
  pasting 177
  positive 162
  precision 23, 108, 185
  present value *See* Investment
  reference *See* Reference
  returning
    DEREF function xxv, 54
    GROWTH function xix, 119-120, 139, 145
    MAX function xix, 35, 149
    MIN function xiii, xix, 152
    overview ix
    TREND function xix, 139, 141, 238-239
  reversing argument value xviii, 160
  selecting xviii, 30, 213-214
  storage in cells 185, 217, 220
  text *See* Text
  types
    selecting 213-214
    testing 135-136
Value axis
  attaching text 19
  scaling 207, 208
#VALUE! error value
  ADD.BAR function 5
  AND function 9
  APP.ACTIVATE function 10
  APP.MOVE function 11
  APPLY.NAMES function 13
  APPLY.STYLE function 14
  ARGUMENT function 15
  ASSIGN.TO.OBJECT function 17
  CELL function 27
  CHOOSE function 30
  COLOR.PALETTE function 33
  CONSOLIDATE function 36
  CREATE.OBJECT function 40, 41
#VALUE! error value, *continued*
  DATA.FORM function 44
  DATEVALUE function 45
  DELETE.ARROW function 51
  DELETE.FORMAT function 52
  DELETE.MENU function 52
  DGET function 57
  DIALOG.BOX function 57
  DUPLICATE function 62
  ENABLE.COMMAND function 67
  EXEC function 70
  EXECUTE function 71
  FIND function 77
  FPOS function 96
  FREAD function 95
  FREADLN function 97
  FSIZE function 98
  FWRITE function 99
  FWRITELN function 99
  GET.CHART.ITEM function 106
  GET.OBJECT function 112, 113
  GROUP function 119
  HIDE.OBJECT function 122
  HLINE function 122
  HLOOKUP function 123
  MDETERM function 149
  MID function 152
  MINVERSE function 153
  MMULT function 155
  OBJECT.PROTECTION function 163
  OFFSET function 163
  ON.TIME function 168
  OR function 172
  PLACEMENT function 182
  POKE function 184
  RENAME.COMMAND function 195
  REQUEST function 198
  returning value type 135
  SAVE.WORKSPACE function 205
  SEARCH function 208
  SELECT function 210





#VALUE! error value, *continued*
  SELECTION function 214
  SEND.TO.BACK function 216
  SHOW.ACTIVE.CELL function 222
  SHOW.DETAIL function 224
  SHOW.LEVELS function 224
  SUBSCRIBE.TO function 230
  SUMPRODUCT function xviii, 232
  TERMINATE function 234
  UNGROUP function 241
  VALUE function xix, 243
  VLOOKUP function 246
  WORKGROUP function 250
VALUE function xix 243
Value-returning functions xxv-xxvi
VAR function xiii, xix, 35, 243, 244
Variable declining balance 244
Variance
  DVAR function xvi, 62
  DVARP function xvi, 62
  VAR function xix, 35, 243, 244
  VARP function xix, 244
VARP function xiii, xix, 244
VDB function xvii, 244-245
Vector xviii, 145-146
Vertical text alignment 88
VIEW.3D function xxi, 245
VLINE function xxiii, 246
VLOOKUP function xviii, 123, 147, 246-247
Volatile functions xxv, 247, 256
VOLATILE function xxv, 247
VPAGE function xxiii, 247
VSCROLL function xxiii, 2, 247-248

## W

WAIT function xxv, 248
WEEKDAY function xvi, 249
WHILE function xxv, 249
WHILE–NEXT loop
  ending xxv, 160
  interrupting xiv, 22
  starting xxv, 249
White Out background 81
Wildcard characters
  deleting files 74, 75
  opening files 169
  replacing text 94
  returning filenames 75
  searching text 208
Window
  activating
    function procedures xxiii, 2-3
    running macros xxv, 168-169
  arranging multiple windows 16
  closing 31-32
  display options xxvi, 115-116
  external application windows 70
  hiding
    HIDE function xxii, 122
    unhiding 241
  Info window
    menu bar ID number 222
    screen display 59, 224
    window display 59-60
  information, returning xxvi, 107, 115-116, 117
  listing open windows xxvi, 250
  maximizing/minimizing
    external application windows 70
    Microsoft Excel window 11
  moving 11-12, 156-157
  names, listing open window xvii, 250
  number 108, 115
  opening 159-160
  pane *See* Pane
  position xxvi, 115-116
  protecting window configuration 188
  restoring to previous size/location 12
  scrolling *See* Scrolling
  sizing *See* Sizing
  splitting xxiii, 97, 116, 227
Window Arrange All command xxii, 16
Window Hide command xxii, 122
Window New Window command xxii, 159
Window Show Clipboard command xxii, 156-157, 223
Window Show Info command xxii, 224
Window Unhide command xxii, 241
Window Workgroup command xxii, 250
WINDOWS function xxvi, 250
WKS file format 169, 204
Workgroup
  creating 250-251
  filling 76
  information, returning 117
WORKGROUP function xxii, 250-251
Worksheet
  calculating *See* Calculating
  clearing 31
  creating 159
  formatting *See* Formatting
  functions *See* Worksheet functions
  gridlines *See* Gridlines
  information, returning xxi, 108-109
  moving objects 84
  multiple worksheets *See* Linking documents
  names, listing all defined 142-143
  pasting 177-179
  printing *See* Printing
  protecting 28, 163, 188-189
  references *See* Reference
  returning number of active worksheets 130
  selecting
    cells xxiii, 209-210, 212-214
    objects xxiii, 211-212
  sizing objects 87-88
Worksheet functions
  arguments xiii
  database functions xvi
  date and time functions xvi
  financial functions xvi-xvii
  information functions xvii

Worksheet functions, *continued*
listing arguments xv
logical functions xviii
lookup functions xviii
mathematical functions xviii
matrix functions xviii
overview ix
statistical functions xix
Worksheet menu bar 5, 222
Workspace
changing settings 251
information, returning xxvi, 116-118
saving 205-206
WORKSPACE function xxii, 251
Wrap text option 9
Write permission 109, 170
Writing to file xiv, 95-96, 99-100

X

XY (scatter) chart 82-84, 85-86

Y

Year
number format, text value equivalent 27
serial number
converting year to xvi, 251-252
translating to date xvi, 44-45
YEAR function xvi, 251-252

Z

Zero 59, 115
Zoom box 98